TEN: Table Explicitization, Neurosymbolically

Anonymous ACL submission

Abstract

We present TEN, a neurosymbolic approach for extracting tabular data from semistructured text such as copy-pasted content from PDFs, emails, or OCR-flattened outputs. This task poses real-world challenges in domains like finance and healthcare, where manual copy-paste into spreadsheets introduces errors and OCR distortions compromise data integrity, leading to financial losses and flawed decisions. Purely neural methods suffer from hallucinations and structural inconsistencies, hindering deployment robustness. TEN addresses this via a novel triadic feedback loop that iteratively refines table hypotheses to enforce constraints and achieve verifiable convergence. Experiments show TEN outperforms neural baselines in exact match accuracy and lower hallucination rates. A 21-participant user study rates TEN's tables more accurate and preferred in over 60% of pairwise comparisons, though verification and correction effort did not differ significantly between conditions.

```
Friday Night

Dugout Box (Behind card's dugout)
Section 143 Row 2 Seat 5-8 $149ea mlb/hub
https://www.ticketmaster.com

Redbird Club (good views, dedicated bathrooms &
concessions)
Section 249 Row 1 Seat 1-10 (aisle) - $58ea mlb/hub
https://www.ticketmaster.com

Mariners Dugout
Section 150 (should look straight in) Row 1
Seat 5-8 (aisle) - $150ea mlb

Cardinals Club
Section 7 Row 1 Seat 1-4 (aisle) - $189ea stubhub
https://www.ticketmaster.com
```

Input text containing seating description

Day	Category	Section	Row	Seats	Price	Section
Friday Night	Dugout Box (Behind card's dugout)	Section 143	Row 2	Seat 5-8	$149	mlb/hub
Friday Night	Redbird Club	Section 249	Row 1	Seat 1-10 (aisle)	$58	mlb/hub
Friday Night	Mariners Dugout	Section 150	Row 1	Seat 5-8 (aisle)	$150	mlb
Friday Night	Cardinals Club	Section 7	Row 1	Seat 1-4 (aisle)	$189	stubhub

Table predicted by LLM Run #1

Day	Section	Details	Price	Platform
Friday Night	Dugout Box (Behind card's dugout)	Section 143, Row 2, Seat 5-8	$149	mlb/hub
Friday Night	Redbird Club (good views...)	Section 249, Row 1, Seat 1-10 (aisle)	$58	mlb/hub
Friday Night	Mariners Dugout	Section 150, Row 1, Seat 5-8 (aisle)	$150	mlb
Friday Night	Cardinals Club	Section 7, Row 1, Seat 1-4	$189	stubhub

Table predicted by LLM Run #2

Figure 1: Comparison of two LLM runs on the identical input. Color coding highlights differing interpretations.

1 Introduction

Knowledge workers devote significant time to manually transferring tabular data from semi-structured documents into spreadsheets for analysis and record-keeping. This process is error-prone and costly: manual entry error rates range from 1% to 5% in typical workflows, rising to 50% for complex data processing (Barchard and Pace, 2011). In clinical settings, transcription errors span 2.3% to 26.9% (Mathes et al., 2017). These errors cascade into financial losses, regulatory risks, and flawed decision-making (Crisanto et al., 2024; Taeihagh, 2025), creating demand for reliable automation in high-stakes domains like finance and healthcare.

Existing approaches fall short on delimiter-free, unstructured text inputs common in enterprise workflows. Layout-based extraction tools such as Tesseract (Smith, 2007) and Tabula (Aristáran et al., 2018) rely on visual metadata (bounding boxes, whitespace delimiters, preserved formatting) and fail when documents are copy-pasted, OCR-flattened, or otherwise linearized into plain text. Large language models (LLMs) offer greater flexibility, demonstrating strong capabilities in natural language understanding and text-to-table conversion (Deng et al., 2024; Andrejczuk et al., 2022). However, purely neural methods suffer from non-determinism and an absence of enforceable structural constraints. Figure 1 shows two runs on identical seating chart data producing divergent table structures (seven vs. five columns), inconsistent labels ("Category" vs. "Section"), and content truncations ("good views..."). This inconsistency demands extensive manual verification and correction in deployment scenarios. Symbolic methods provide interpretability, but their rigidity makes them brittle to irregular delimiters, inconsistent spacing, or nested structures. Thus, a key gap persists: current methods lack a hybrid approach that merges neural adaptability with symbolic constraints to prevent hallucinations, and structural errors.

We introduce TEN, an iterative neurosymbolic framework that bridges this gap through verifiable refinement. TEN operates in a triadic feedback loop: (1) an LLM generator produces initial table hypotheses, (2) a deterministic symbolic checker detects violations, and (3) a critique-LLM translates checker feedback into natural language refinements. This triadic architecture differs from prior two-stage feedback systems, where a generator and critic exchange direct natural-language feedback. In contrast, TEN inserts an intermediate deterministic symbolic validation layer that outputs constraint-grounded violation signals, which a critique-LLM then translates into contextualized refinements for the generator. This design keeps feedback formally verifiable while remaining task-adaptive, helping to reduce common issues such as over-correction, hallucinated critiques, and subjective termination. Also, the symbolic layer supports deterministic convergence as iteration stops only when all constraints are fully satisfied.

Our contributions are threefold: **(1)** We present TEN, a novel neurosymbolic framework that enables verifiable and reliable table extraction from delimiter-free, semistructured text by combining neural flexibility with symbolic enforceability, overcoming core limitations of prior neural-only and symbolic-only methods. **(2)** We show that Structural Decomposition prompting outperforms standard COT and naive extraction baselines for initial table hypotheses, and that our symbolic-guided critique refinement targets corrections more effectively than LLM-only or symbolic-only strategies. **(3)** We show through empirical evaluation that TEN-generated tables achieve statistically significant improvements in accuracy ratings (5.0 vs. 4.3) and are preferred in over 60% of cases for verification and correction tasks.

2 Related Work

Table Extraction from Unstructured Sources. Early methods rely on visual layout heuristics and spatial clustering (e.g., Tabula (Aristáran et al., 2018), PDFMiner (Shinyama, 2014), TableNet (Paliwal et al., 2019), LayoutLMv2 (Xu et al., 2021)), supported by datasets like PubTabNet (Zhong et al., 2019) and TableBank (Li et al., 2019). These perform well on formatted documents but fail on linearized, delimiter-free text from copy-paste or OCR flattening. Recent text-based approaches include seq2seq models (Wu et al., 2022), tuple organization (Deng et al., 2024), and neurosymbolic systems like Revilio (Singh et al., 2024) (LLM header/sketch + enumerate-and-test). TEN advances this line with an iterative triadic loop: LLM hypothesis generation, symbolic validation for precise violation signals, and critique-driven refinement. Unlike single-pass or enumerate-test methods, this ensures robust, verifiable convergence on noisy inputs, enabling reliable production deployment.

Table Structure Recovery. Table understanding infers headers, entities, alignment, and roles via rule-based cues (Dong et al., 2019) or contextual models (e.g., TaBERT (Yin et al., 2020), TAPAS (Herzig et al., 2020), TURL (Deng et al., 2020)). Recent efforts frame it as sequence labeling or pointer generation (Khang and Hong, 2025). TEN focuses on noisy, malformed extractions by iteratively inducing structure: the symbolic checker flags inconsistencies, and the critique-LLM generates coherent fixes. This avoids assumptions of clean inputs and supports incremental refinement on delimiter-free text.

Vision-Language Models (VLMs). Recent VLMs like PaliGemma 2 (**?**) perform well on table extraction from images by leveraging layout cues. TEN addresses the complementary text-only regime, e.g., pasted email/chat/web content, privacy-constrained pipelines forbidding image APIs, legacy systems, and screen-reader accessibility, where structure must be inferred purely from tokens and delimiters, without visual information.

Iterative Self-Refinement in LLMs. Iterative refinement advances LLM performance through self-feedback (Madaan et al., 2023), reflection (Shinn et al., 2023), COT (Wei et al., 2022), TOT (Yao et al., 2023), or external critique (Gou et al., 2024). Methods like LLMRefine (Xu et al., 2024) use LLM-generated feedback, while others

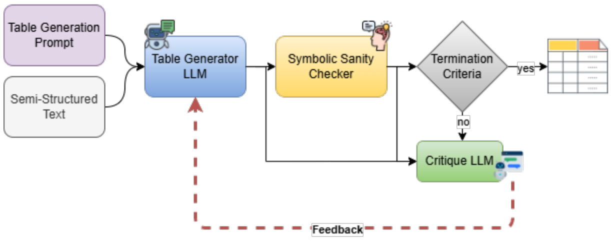


Figure 2: Overview of TEN.

integrate symbolic tools. Pure LLM feedback often remains generic or biased. TEN adapts this paradigm to table extraction from noisy text via a triadic loop.

3 The Table Explicitization Problem

The goal of table explicitization is to transform an unstructured or semi-structured input string T, lacking explicit row/column delimiters, into a reliably structured format T'. This is difficult due to noisy, irregular formatting from copy-paste, OCR-flattened PDFs, emails, and reports, which causes traditional parsers to fail. For example, as shown in Figure 1, a copied ticket seating description loses delimiters and mixes hierarchical attributes (sections, pricing, notes, URLs) across irregular lines. The explicitized output T' recovers the intended tabular structure in clean JSON (partial example:)

```
{
  "Dugout Box": {
    "Section": "143",
    "Row": "2",
    "Seats": "5-8",
    "Price": "$149ea",
    "Source": "mlb/hub",
    "Link": "https://www.ticketmaster.com",
    "Notes": "Behind card's dugout"
  },{ ... }
}
```

4 The TEN Approach

As shown in Figure 2, TEN uses an iterative neurosymbolic loop: a Table Generator LLM drafts via Structural Decomposition, a Symbolic Sanity Checker detects precise violations, and a Critique LLM applies targeted refinements, enforcing hard constraints and converging to high-fidelity tables, outperforming one-shot baselines on noisy inputs.

4.1 Structural Decomposition Prompting

Structural Decomposition Prompting is a specialized OT technique that instructs the LLM to first identify table-like regions in the input, detect natural delimiters and boundaries, and plan row separators before generating any structured output. This two-stage reasoning, segmentation followed by explicit tabular reconstruction, mimics human table parsing and leverages the model's holistic contextual understanding to avoid brittle, hand-crafted layout heuristics.

The prompt emphasizes high recall in the initial phase, capturing all plausible tabular content without hallucinating new entries. The resulting partial tables are then unified into a single structured representation. By prioritizing boundary detection and semantic coherence early, this design substantially improves robustness on noisy, delimiter-free text. See Appendix F for an example on PDF-extracted content and Appendix G.1 for the full prompt.

4.2 Symbolic Sanity Checker

Symbolic sanity checker applies a set of handcrafted and domain-agnostic symbolic rules to assess the quality of a given table. The input to the sanity checker is the predicted table and the input text and it returns table goodness/badness score.

Entity Consistency Rules: For each row/column, this rule checks if all data values in that row/column match a entity regular expression. We use a fixed collection of entities, such as date, time, email, url, and word.

Signature-Based Column Analysis: When a column does not contain a predefined entity, we use syntactic features, such as presence or absence of special characters (digits, letters, punctuations) to detect inconsistencies, such as text in a numeric column or unexpected delimiters.

Merged Cell Detection: A common and subtle error is the merging of adjacent cells, where column boundaries are lost and values collapse into one cell. Though no data is lost, column alignment is distorted. We detect such cases by flagging cells with multiple numeric tokens (e.g., "102, 205") or mixed formats (e.g., "Revenue 750").

Delimiter-Induced Errors: Another frequent table extraction error arises from using the wrong column separator; for example, the number "12,345" splitting into "12" and "345". Such issues often evade column-wise checks, so we detect them by analyzing adjacent row cells for numeric fragmentation patterns.

Parenthesis and Bracket Matching: Mismatched brackets are indicators of corrupted cell content. To catch such anomalies, we scan and

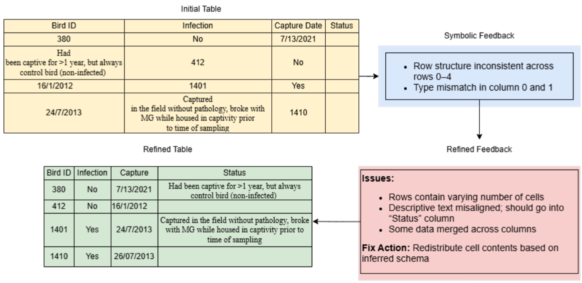


Figure 3: Example of table generation and refinement in TEN. Initial table has structural inconsistencies. Feedback loop identifies them and infers a corrected structure, triggering a refinement step.

flag all cells containing unbalanced opening and closing symbols.

Empty Row Detection: Rows with only whitespace or empty strings are identified and marked, as they usually indicate a malformed table region.

The sanity checker aggregates the findings from the above rules into two metrics: **goodness** score is the proportion of cells in the full table that have some desirable property, and **badness** score is the maximum proportion of cells in a column that have some undesirable property (possible merged cell, incorrect delimiter, etc.). The rule violations are noted and returned, and subsequently forwarded to the critique LLM. In addition to goodness and badness scores, the sanity checker also computes **coverage**, and **hallucination rate**. These scores are used to decide if the feedback loop termination.

4.3 Critique LLM

While the symbolic checker detects structural issues (e.g., uneven row lengths and merged cells in Figure 3), its reliance on surface patterns can produce false positives on valid but atypically formatted tables. Passing unfiltered feedback directly to regeneration risks over-correction. To mitigate this, we introduce a Critique LLM as a reasoning-aware intermediary. It interprets the table and symbolic violations, filtering out spurious signals and decides whether to accept/refine/disregard feedback.

This completes one full iteration of TEN's neurosymbolic loop: initial generation, symbolic checking, contextual critique, and targeted regeneration. The process repeats until the sanity checker approves the table or a maximum iteration limit is reached. See Appendix A for the algorithm.

Dataset	# Tables	# Tokens	# Rows	# Cols
FinTabNet	1000	111.41	13.54	2.59
PubTabNet	1000	120.3	14.34	4.09
BrokenCSV	1000	128734.3	14871.65	148.65
WikiTables	1000	219.81	16.06	6.2
FinReCon20	20	217.55	52.1	4.7

Table 1: Dataset Statistics.

5 Evaluation

Datasets. We evaluate TEN on benchmarks with diverse structural complexity, noise levels, and semantics (Table 1): PubTabNet (Zhong et al., 2019) and FinTabNet (Zheng et al., 2020) (using raw OCR text), WikiTables (Bhagavatula et al., 2013), BrokenCSV (van den Burg et al., 2019), and FinReCon20, a 20-table set, manually curated from real financial documents for user study.

Baselines. We compare against six LLMs: **GPT-4**, **DeepSeek-R1**, **o3-mini**, **DeepSeek-V3-0324**, **Phi-4**, and **Mistral-Small-2503**. We also include two table reconstruction baselines: the neurosymbolic Tabularis Revilio (Singh et al., 2024) and the symbolic SplitText (Raza and Gulwani, 2017, 2020). Narrative-style text-to-table methods (Deng et al., 2024; Wu et al., 2022) are excluded, as their inputs differ from our flattened, noisy text.

Evaluation Metrics. We use both structural and semantic metrics: **Coverage**: fraction of input alphanumeric characters preserved in the output $(1 - \frac{|U|}{|S|})$. **Hallucination Rate**: average fraction of unmatched cell content, cell-wise then row-averaged. **Exact Match (EM)**: binary score for perfect structure and content match to ground truth. **Tree Edit Distance (TED)**: normalized cell-level edit distance (Zhang and Shasha, 1989). **Column Match (Col.V.M.)**: average percentage of exactly matched columns (micro-averaged). **Value Match (C.V.M.)**: average percentage of correctly reconstructed cell values.

User Study. We conducted a within-subjects study with 21 active spreadsheet users across 2–4 sessions each. Participants viewed a ground-truth table and two predicted tables (from different tools) extracted from its textual form, then partially edited the predictions to match the ground truth (sessions video-recorded). They rated perceived accuracy, verification effort, and correction effort via questionnaire (see Appendices B and E for details and questionnaire).

Method	Metric	PubTabNet	FinTabNet	WikiTables	BrokenCSV
TEN	TED	0.67	0.47	0.65	0.22
	EM	61.60	21.7	36.80	81.6
	C.V.M.	71	33	64	81
	Col.V.M.	75	29	56	69
Revilio	TED	0.60	0.53	0.59	0.17
	EM	52.34	15.65	33.43	60.96
	C.V.M.	66	29	59	76
	Col.V.M.	60	23	51	63
SplitText	TED	0.55	0.58	–	0.43
	EM	26	11	1.1	53.8
	C.V.M.	37	22	1	46
	Col.V.M.	35	19	8	40

Table 2: TEN (GPT-4o) vs. existing baselines.

6 Results and Analysis

RQ1: *Influence of prompting strategy and model choice on performance?*
Structure first "beats reason more": We compared Structural Decomposition (SD) prompting against baseline (direct table construction) and CoT across four datasets and six models. Figure 4 shows SD outperforms both by 10-40 EM points and 8-15 in ColVM/CVM. CoT yields minor gains over baseline. Full results in Appendix C.
Model Capacity Matters: Larger models excel across strategies, but SD narrows the gap, e.g., o3-mini gains 6× EM on FinTabNet (2.2% to 14.4%), GPT-4 2.4× (8.9% to 21.7%).
Large tables are difficult: EM drops with more rows (Figure 5) and peaks at 6-8 columns before declining due to alignment ambiguities. SD and larger models mitigate this.
Dataset Complexity: Based on LLM performance, the datasets can be ordered in difficulty level as BrokenCSV < PubTabNet < WikiTables < FinTabNet. We observe that the "easier" datasets have contextual information that make it easier to recover structure even when spatial and visual layout information is not present.
Iterative Feedback Loop Performance: We observe the largest gains in EM, CVM and ColVM metrics in the first two to three iterations, consistent with several past works (Figure 7 in Appendix C).
Computational Requirements: Processing a table with TEN takes 5-15 seconds on average, consuming approximately 6K-15K tokens depending on table complexity. Most tables converge within 2-3 iterations. At standard API rates, this translates to roughly $0.08-0.15 per table.
Robustness: Statistical tests confirm significance (Appendix C).

RQ2: TEN *comparison to baselines.*

Table 2 shows that TEN consistently outperforms Revilio and SplitText on content-fidelity metrics (EM, CVM, ColVM) across all four benchmarks. On PubTabNet, TEN improves EM by +9.26 points over Revilio, with gains of +5 in CVM and +15 in ColVM. On the more difficult FinTabNet, where absolute EM is lower for all methods, TEN still achieves a +6.05-point EM gain along with higher CVM and ColVM. On WikiTables, improvements are smaller but consistent (+3.37 EM, +5 CVM, +5 ColVM). The largest gains occur on BrokenCSV, where TEN reaches 81.6 EM (vs. 60.96 for Revilio and 53.8 for SplitText), with better CVM and ColVM. On structural alignment (TED), results are mixed. TEN sometimes normalizes header layouts, consolidates multi-row headers, flattens/propagates spans, removes empty scaffold rows/columns, or relocates footnotes and units, preserving all content values while changing the parse tree. These benign normalizations improve content-fidelity metrics but can modestly increase TED, which strictly rewards structural isomorphism and penalizes even harmless layout differences. TED should therefore be interpreted together with content-fidelity metrics rather than in isolation.

RQ3: *Is* TEN *useful in practice?*
We conducted a user study where participants were given tables generated by two variants of TEN: original TEN and a slightly weaker version of TEN where we use CoT prompt that we call TENCOT. Figure 6 summarizes participants' responses to the different questions in the questionnaire provided to them. We found that on the Likert scale (1-7), participants reported that the accuracy of tables generated by TEN was significantly higher than of those generated by TENCOT (Wilcoxon signed-rank test, $W = 284.0, p = .02$) with a moderate effect size ($r = -0.30$). Further, in 7 out the 20 tables used for the study, all three reviewers independently rated TEN to be more accurate than TENCOT, while all participants preferred TENCOT for only 1 table. Despite this difference in responses on the tables' accuracy, we did not observe any significant differences in participants' responses on the effort required to verify ($W = 260.0, p = .51$) or fix ($W = 266.0, p = .08$) these tables. This might be as while TEN required users to often fix alignment issues, TENCOT often required users to add missing data, both of which led to increased effort in fixing the replication (details in Appendix C).

RQ4: *Impact of different components of* TEN.

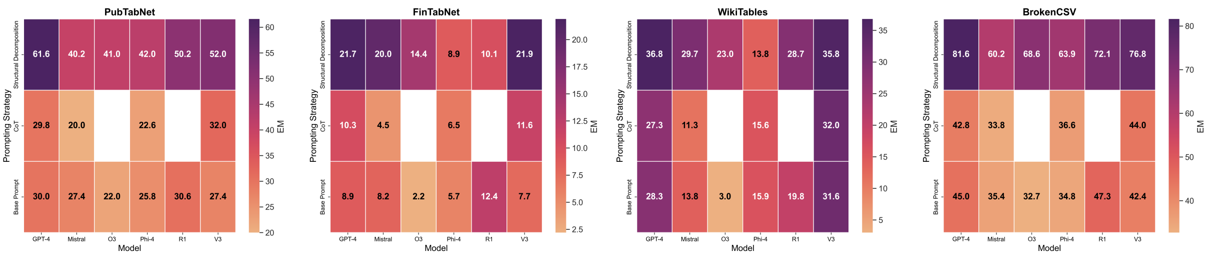


Figure 4: Exact Match (EM) accuracy (darker is better) by prompting strategy and model on four datasets.

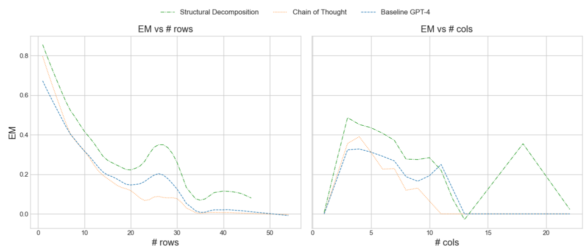


Figure 5: EM accuracy vs. table size

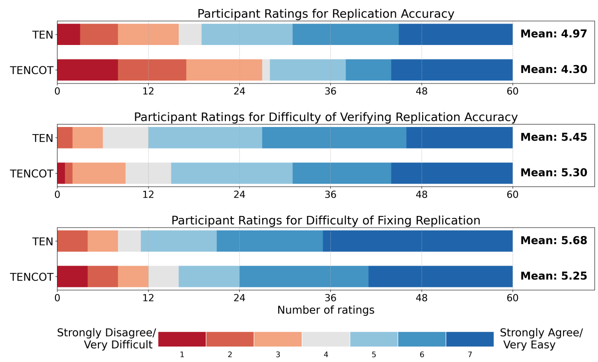


Figure 6: Questionnaire responses in user study.

Feedback	Metric	Wikipedia	BrokenCSV	PubTabNet	FinTabNet
No	Coverage	0.92	0.94	0.93	0.90
	Halluc.	0.13	0.17	0.18	0.17
	TED	0.71	0.36	0.76	0.50
	EM	15.18	60.31	17.16	12.91
	C.V.M.	42.31	65.34	54.98	22.96
	Col.V.M.	45.49	61.01	48.12	19.26
Symbolic	Coverage	0.94	0.96	0.93	0.91
	Halluc.	0.11	0.14	0.15	0.15
	TED	0.68	0.30	0.75	0.48
	EM	20.36	68.93	21.28	20.34
	C.V.M.	55.84	68.56	59.96	25.36
	Col.V.M.	57.69	63.35	53.35	21.12
LLM	Coverage	0.94	0.96	0.94	0.92
	Halluc.	0.11	0.15	0.11	0.11
	TED	0.66	0.31	0.73	0.48
	EM	27.93	72.45	22.83	18.69
	C.V.M.	59.01	74.93	55.69	29.12
	Col.V.M.	54.69	70.89	52.87	24.95
Both	Coverage	0.94	0.97	0.93	0.91
	Halluc.	0.11	0.15	0.14	0.15
	TED	0.65	0.28	0.67	0.47
	EM	36.80	81.6	61.60	21.73
	C.V.M.	64.36	81.03	71.54	33.92
	Col.V.M.	59.59	69.56	75.69	29.36

Table 3: Ablation Study for Variants of TEN.

To evaluate TEN 's feedback mechanisms, we ablated its components and compared four variants under identical prompts, and evaluation setup: (i) No feedback; (ii) Symbolic feedback only; (iii) LLM feedback only; (iv) **Both** (full TEN). Table 3, Symbolic feedback alone consistently improves structural and content metrics, with EM gains of +5.18 (Wikipedia) and +8.62 (BrokenCSV). LLM feedback provides even larger gains on simpler or noisier datasets, e.g., +12.75 EM (Wikipedia). However, on complex tables, it is less stable and may underperform Symbolic feedback. The full system, **Both**, achieves the best performance across all metrics. It offers the highest EM, CVM, lowest TED, and near-saturated coverage (0.90–0.97). In some cases, LLM feedback slightly reduces hallucination compared to the full system, but this comes at the cost of structural fidelity.

Case Study Examples. In a PubTabNet clinical table (App. H.1), LLM-only feedback hallucinated a column from misaligned headers; NeuroSymbolic feedback corrected alignment and promoted labels, boosting EM and CVM with minimal TED impact. In another example (App. H.2), LLM-only feedback, flattened hierarchical group headers into flat columns (improving readability but raising TED); NeuroSymbolic feedback preserved structure, yielding higher fidelity and lower overall error.

7 Conclusion

We present TEN, a neurosymbolic framework that recovers high-fidelity tables from noisy, unstructured text via an iterative triadic feedback loop. By enforcing hard symbolic constraints without visual layout or external images, TEN substantially outperforms one-shot and existing text-only baselines across diverse real-world scenarios.

References

Ewa Andrejczuk, Julian Martin Eisenschlos, Francesco Piccinno, Syrine Krichene, and Yasemin Altun. 2022. Table-to-text generation and pre-training with tabt5. *Preprint*, arXiv:2210.09162.

Manuel Aristáran, Mike Tigas, Jeremy B. Merrill, and Jason Das. 2018. Tabula: Extract tables from pdfs. Open-source tool, https://tabula.technology/.

Kimberly A. Barchard and Larry A. Pace. 2011. Preventing human error: The impact of data entry methods on data accuracy and statistical results. *Computers in Human Behavior*, 27(5):1834–1839. 2009 Fifth International Conference on Intelligent Computing.

Chandra Sekhar Bhagavatula, Thanapon Noraset, and Doug Downey. 2013. Methods for exploring and mining tables on wikipedia. In *Proceedings of the ACM SIGKDD Workshop on Interactive Data Exploration and Analytics*, IDEA '13, page 18–26, New York, NY, USA. Association for Computing Machinery.

Juan Carlos Crisanto, Cris Benson Leuterio, Jermy Prenio, and Jeffery Yong. 2024. Regulating ai in the financial sector: recent developments and main challenges. Fsi insights on policy implementation; no. 63, Financial Stability Institute, Bank for International Settlements.

Xiang Deng, Huan Sun, Alyssa Lees, You Wu, and Cong Yu. 2020. TURL: table understanding through representation learning. *CoRR*, abs/2006.14806.

Zheye Deng, Chunkit Chan, Weiqi Wang, Yuxi Sun, Wei Fan, Tianshi Zheng, Yauwai Yim, and Yangqiu Song. 2024. Text-tuple-table: Towards information integration in text-to-table generation via global tuple extraction. *Preprint*, arXiv:2404.14215.

Haoyu Dong, Shijie Liu, Zhouyu Fu, Shi Han, and Dongmei Zhang. 2019. Semantic structure extraction for spreadsheet tables with a multi-task learning architecture. In *Workshop on Document Intelligence at NeurIPS 2019*.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. 2024. Critic: Large language models can self-correct with tool-interactive critiquing. *Preprint*, arXiv:2305.11738.

Jonathan Herzig, Pawel Krzysztof Nowak, Thomas Müller, Francesco Piccinno, and Julian Martin Eisenschlos. 2020. TAPAS: weakly supervised table parsing via pre-training. *CoRR*, abs/2004.02349.

Minsoo Khang and Teakgyu Hong. 2025. Tflop: Table structure recognition framework with layout pointer mechanism. *Preprint*, arXiv:2501.11800.

Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, Ming Zhou, and Zhoujun Li. 2019. Tablebank: Table benchmark for image-based table detection and recognition. *CoRR*, abs/1903.01949.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. 2023. Self-refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems*, volume 36, pages 46534–46594. Curran Associates, Inc.

Tim Mathes, Pauline Klaßen, and Dawid Pieper. 2017. Frequency of data extraction errors and methods to increase data extraction quality: a methodological review. *BMC Medical Research Methodology*, 17(1):152.

Quinn McNemar. 1947. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157.

Shubham Singh Paliwal, Vishwanath D, Rohit Rahul, Monika Sharma, and Lovekesh Vig. 2019. Tablenet: Deep learning model for end-to-end table detection and tabular data extraction from scanned document images. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 128–133.

Mohammad Raza and Sumit Gulwani. 2017. Automated Data Extraction Using Predictive Program Synthesis. In *AAAI*, pages 882–890.

Mohammad Raza and Sumit Gulwani. 2020. Web data extraction using hybrid program synthesis: A combination of top-down and bottom-up inference. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, pages 1967–1978.

Jeff Sauro and Joseph S. Dumas. 2009. Comparison of three one-question, post-task usability questionnaires. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '09, page 1599–1608, New York, NY, USA. Association for Computing Machinery.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik R Narasimhan, and Shunyu Yao. 2023. Reflexion: language agents with verbal reinforcement learning. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Yusuke Shinyama. 2014. Pdfminer: Python pdf parser and analyzer. `https://github.com/euske/pdfminer`. Accessed: 2025-04-17.

Mukul Singh, Gust Verbruggen, Vu Le, and Sumit Gulwani. 2024. Tabularis revilio: Converting text to tables. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, CIKM '24, page 4056–4060, New York, NY, USA. Association for Computing Machinery.

Ray Smith. 2007. An overview of the tesseract ocr engine. In *Proceedings of the Ninth International Conference on Document Analysis and Recognition (ICDAR)*, pages 629–633. IEEE.

Araz Taeihagh. 2025. Governance of generative ai. *Policy and Society*, 44(1):1–22.

G. J. J. van den Burg, A. Nazábal, and C. Sutton. 2019. Wrangling messy csv files by detecting row and type patterns. *Data Min. Knowl. Discov.*, 33(6):1799–1820.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed H. Chi, Quoc Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. *CoRR*, abs/2201.11903.

Xueqing Wu, Jiacheng Zhang, and Hang Li. 2022. Text-to-table: A new way of information extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2518–2533, Dublin, Ireland. Association for Computational Linguistics.

Wenda Xu, Daniel Deutsch, Mara Finkelstein, Juraj Juraska, Biao Zhang, Zhongtao Liu, William Yang Wang, Lei Li, and Markus Freitag. 2024. LLMRefine: Pinpointing and refining large language models via fine-grained actionable feedback. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 1429–1445, Mexico City, Mexico. Association for Computational Linguistics.

Yang Xu, Yiheng Xu, Tengchao Lv, Lei Cui, Furu Wei, Guoxin Wang, Yijuan Lu, Dinei Florencio, Cha Zhang, Wanxiang Che, Min Zhang, and Lidong Zhou. 2021. LayoutLMv2: Multi-modal pre-training for visually-rich document understanding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2579–2591, Online. Association for Computational Linguistics.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik R Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Pengcheng Yin, Graham Neubig, Wen-tau Yih, and Sebastian Riedel. 2020. Tabert: Pretraining for joint understanding of textual and tabular data. *CoRR*, abs/2005.08314.

Kaizhong Zhang and Dennis Shasha. 1989. Simple fast algorithms for the editing distance between trees and related problems. *SIAM Journal on Computing*, 18(6):1245–1262.

Xinyi Zheng, Douglas Burdick, Lucian Popa, and Nancy Xin Ru Wang. 2020. Global table extractor (GTE): A framework for joint table identification and cell structure recognition using visual context. *CoRR*, abs/2005.00589.

Xu Zhong, Elaheh ShafieiBavani, and Antonio Jimeno Yepes. 2019. Image-based table recognition: data, model, and evaluation. *arXiv preprint arXiv:1911.10683*.

A TEN's Algorithm

Algorithm 1 outlines the main pipeline, which iteratively generates tables using Structural Decomposition prompting, validates them with symbolic rules, and refines them via critique feedback. The process initializes an empty table, generates an initial structure, and iterates until convergence or a maximum number of iterations is reached, guided by metrics such as coverage and hallucination rate. The `SymbolicSanityChecker` (Algorithm 2) is a critical component, applying domain-agnostic rules to detect well formedness of a table. It computes metrics (e.g., goodness and badness scores) to inform refinement, ensuring robust table structures. Detailed prompts for table generation and critique are provided in Appendices G.1, G.3 and G.2, respectively.

B Evaluation

The goal of our evaluation is to assess the effectiveness of TEN in producing high-fidelity, structurally coherent tables from noisy and unstructured text. Specifically, our evaluation investigates two key objectives. **First**, we examine whether the integration of symbolic sanity checks and critique-guided regeneration leads to improved table quality by reducing hallucinations, and correcting structural inconsistencies. **Second**, we compare TEN against symbolic-only pipelines and neural-only(LLM-based) baselines to highlight the advantages of combining rule-based structural validation with neural generation in an iterative refinement framework. **Third**, we conduct a mixed-methods empirical study with spreadsheet users to understand the practical usefulness and drawbacks of tables generated by TEN.We aim to answer the following research questions:

RQ1: To what degree do prompting strategies and model choice influence table-extraction performance when evaluated on datasets of escalating complexity?

RQ2: How accurately does TEN reconstruct tables compared to existing baselines?

RQ3: How useful is TEN at reconstructing tables from real-world financial documents?

RQ4: What is the impact of different components of TEN?

B.1 Datasets

To assess TEN, we chose benchmarks with varied structural complexity, formatting noise, and seman-

Algorithm 1 TEN: Table Explicitization, Neurosymbolically

Require: Unstructured text T, max iterations N, convergence threshold C
Ensure: Structured table T'

```
Initialize iteration_count ← 0
Initialize table_candidate ← ∅                                   ▷ Empty table structure
Initialize critique_feedback ← ∅                                 ▷ No feedback for first iteration
repeat
    iteration_count ← iteration_count + 1
    if iteration_count = 1 then
        table_candidate ← STRUCTURALDECOMPOSITIONPROMPTING(T) ▷ Identifies table-like regions, Appendix G.1
    else
        table_candidate ← REFINETABLE(T, table_candidate, critique_feedback) ▷ Refines table based on feedback,
Appendix G.3
    end if
    validation_results ← SYMBOLICSANITYCHECKER(table_candidate, T)
    if ISCONVERGED(validation_results, C) then
        break
    end if
    critique_feedback ← CRITIQUELLM(table_[candidate], validation_results, T) ▷ Generates actionable feedback,
Appendix G.2,
until iteration_count ≥ N
return table_candidate
```

tic richness, summarized in Table 4.

PubTabNet: The PubTabNet dataset (Zhong et al., 2019) contain tables from open-source scientific articles with OCR annotations. We generate benchmark tasks by using the raw OCR output as unstructured table text to simulate real-world reconstruction challenges.

FinTabNet: The FinTabNet dataset (Zheng et al., 2020) consists of tables extracted from annual reports of S&P 500 companies. Each table is annotated with fine-grained cell structure information obtained via token alignment between PDF and HTML versions of the documents. Similar to PubTabNet, we use the raw OCR text of each table to generate unstructured inputs for our benchmark.

Wikipedia Tables: *WikiTables* dataset (Bhagavatula et al., 2013) comprises structured tables from Wikipedia with clear formatting and rich semantic content. Despite their structural simplicity, these tables often rely heavily on positional context, making them challenging for unstructured text processing. We convert each table into unstructured OCR-style text, following the same procedure used for PubTabNet.

BrokenCSV: The BrokenCSV dataset (van den Burg et al., 2019), consists of noisy and irregular CSV-formatted tables with inconsistent delimiters, and missing entries.

FinRecon20: While prior benchmarks focus on structure extraction from images, our task centers on a different challenge: *reconstructing structured tables from free-form text*. To evaluate this, we introduce FinReCon20, a curated dataset of 20 tables manually extracted from real-world financial documents, including balance sheets, earnings reports, and disclosures. These tables were selected to reflect common copy-paste issues seen in document workflows, such as inconsistent delimiters, fragmented rows, multi-line cells, and token-level segmentation errors. Dataset details appear in Table 4.

Dataset	# Tasks	# Tokens	# Rows	# Columns
FinTabNet	1000	111.41	13.54	2.59
PubTabNet	1000	120.3	14.34	4.09
BrokenCSV	1000	128734.3	14871.65	148.65
Wikipedia Tables	1000	219.81	16.06	6.2
FinReCon20	20	217.55	52.1	4.7

Table 4: Dataset statistics used in our experiments. We report the number of tasks, and average number of tokens, rows, and columns per table.

B.2 Baselines

We evaluate TEN using six representative language models: GPT-4, DeepSeek-R1, o3-mini, DeepSeek-V3-0324, Phi-4, and Mistral-Small-2503. Each model is tested under three prompting conditions to assess the robustness of TEN structural decomposition prompting. The first condition employs a baseline prompt that simply instructs the model to construct a table from unstructured text. The second condition uses Chain-of-Thought(CoT) prompting (Wei et al., 2022), introducing intermediate reasoning by appending "let's think step by step" to the instruction. The third condition applies our proposed structural decomposition prompting strat-

Algorithm 2 SymbolicSanityChecker

```
procedure SymbolicSanityChecker(table_candidate, T)
    Initialize violations ← ∅, goodness_score ← 0, badness_score ← 0
    Initialize coverage ← 0, hallucination_rate ← 0
    total_cells ← CountCells(table_candidate)
    consistent_cells ← 0, violating_cells ← 0
  ▷ Empty Row Detection
    for each row r in table_candidate do
        if IsEmptyRow(r) then                                    ▷ Whitespace or empty strings
            Add "Empty Row" to violations
            violating_cells ← violating_cells + CountCells(r)
        end if
    end for
  ▷ Entity Consistency and Column Analysis
    for each column c in table_candidate do
        entity_type ← DetectEntityType(c)
        if MatchesEntityType(c, entity_type) then
            consistent_cells ← consistent_cells + CountCells(c)
        else
            inconsistencies ← AnalyzeColumnSyntax(c)
            if inconsistencies ≠ ∅ then
                Add "Inconsistent Column" to violations
                violating_cells ← violating_cells + CountInconsistentCells(c)
            end if
        end if
    end for
  ▷ Merged Cell, Delimiter, and Bracket Checks
    for each cell cell in table_candidate do
        if HasMultipleEntityTokens(cell) then
            Add "Merged Cell" to violations
            violating_cells ← violating_cells + 1
        end if
        if HasUnbalancedBrackets(cell) then
            Add "Unbalanced Brackets" to violations
            violating_cells ← violating_cells + 1
        end if
        if HasDelimiterErrors(cell) then
            Add "Delimiter Error" to violations
            violating_cells ← violating_cells + 1
        end if
    end for
  ▷ Compute and Return Metrics
    coverage ← CountTokensInTable(T, table_candidate) / CountTokens(T)
    non_input_tokens ← CountTokensNotInInput(T, table_candidate)
    total_table_tokens ← CountTokens(table_candidate)
    hallucination_rate ← non_input_tokens / total_table_tokens
    goodness_score ← consistent_cells / total_cells
    badness_score ← violating_cells / total_cells
    return {
      violations,
      coverage,
      hallucination_rate,
      goodness_score,
      badness_score
    }
end procedure
```

egy, which explicitly segments the task into logical subcomponents to guide model behavior.

In addition to these prompting variants, we compare against two baselines designed for structured table reconstruction. The first is Tabularis Revilio (Singh et al., 2024), a neurosymbolic pipeline developed to convert flattened tabular text into accurate table representations. The second baseline is SplitText, a method for table reconstruction that segments flattened tabular content into columns by leveraging linguistic and statistical cues (Raza and Gulwani, 2017, 2020).

While TEN may seem related to prior models such as (Deng et al., 2024; Wu et al., 2022), their input settings and goals differ substantially from ours. TEN assumes that the input is a semistructured textual representation of some table. In contrast, the input for these other systems is a narrative-style paragraph description that contains some information that can be extracted as a table. Since the input type is different, we do not include these models as baselines.

B.3 Evaluation Metrics

We adopt both structural and semantic metrics to comprehensively evaluate the quality of the generated tables. **Coverage** is defined as the fraction of alphanumeric characters from the input that are preserved in the output table: **Coverage** $= 1 - \frac{|U|}{|S|}$ where $|S|$ is the number of alphanumeric characters in the source input, and $|U|$ is the number of uncovered characters based on string matching. **Hallucination Rate** is defined as the average fraction of unmatched content across all cells in the generated table. For a table with R rows, each containing C_i cells, it is computed as: **HallucinationRate** $= \frac{1}{R}\sum_{i=1}^{R}\left(\frac{1}{C_i}\sum_{j=1}^{C_i}(1 - \text{cellCoverage}_{ij})\right)$, where $cellCoverage_{ij} \in [0, 1]$ denotes the degree to which cell(i,j) matches the input source(including partial matches). **Exact Match(EM)** is a binary indicator that evaluates to 1 if the predicted table matches the ground-truth table exactly in both structure and content, and 0 otherwise. **Tree Edit Distance (TED)** is computed as the normalized cell-level edit distance between the predicted and ground-truth tables (Zhang and Shasha, 1989). **Column Match (Col.V.M.)** is defined as the average percentage of columns that are exactly matched between the predicted and reference tables, micro-averaged per table and then averaged across all tables. **Value Match (C.V.M.)** reports the average percentage of individual cell values that are correctly reconstructed across all tables.

B.4 User Study

We conducted a within-subject empirical study with 21 spreadsheet users in which each participant evaluated the accuracy of and corrected errors in tables generated by both TEN (Structural Decomposition), hereafter referred to as TEN, and those generated by TEN (COT), hereafter referred to as TENCOT. This design allowed us to assess the practical strengths and limitations of these different approaches to table extraction and to identify which types of errors users found most difficult to fix and verify.

Tasks: Each participant worked on 2–4 tables from the FinReCon20 dataset[1], using Microsoft Excel to assess and fix outputs produced by both TEN and TENCOT in a within-subject design. To control for individual biases, each table was independently reviewed by three different participants. The order of tool outputs was randomized to mitigate ordering effects.

In cases where table fixes involved repetitive or tedious edits (e.g., adding multiple rows of similar data), participants were instructed to perform only a representative sample of the changes. This approach helped us gauge the perceived difficulty and effort without requiring exhaustive correction. After completing each task, participants filled out a questionnaire addressing (1) perceived accuracy of the table, (2) effort required to verify its correctness, and (3) effort required to make corrections.

Participants: We recruited 21 active spreadsheet users through an online user study recruitment platform. To ensure sufficient proficiency, we only included participants who reported using Microsoft Excel on at least 8 or more days in the past month. Given the financial nature of our dataset, we also required participants to have some experience working with financial tables in Excel (all participants reported working with financial spreadsheets at least occasionally). However, we did not restrict participation to individuals in finance-related occupations, allowing us to recruit a diverse pool of users across domains. This ensured a realistic mix of spreadsheet users while maintaining a baseline familiarity with financial tabular data.

Study Protocol: All study sessions were conducted by one of the authors. At the beginning of

[1] https://anonymous.4open.science/r/FinRecon20-Dataset-1B11/

each session, the administrator briefed participants on the task they would be performing, followed by demonstrating a tutorial using an example table. In pilot studies, we observed that participants would often focus on the formatting and visual aspects (e.g., font and spacing) and the robustness (e.g., number of formulas) of the replications when evaluating and fixing them. Since neither TEN nor TENCOT currently implement such aspects, the study administrator asked participants to focus solely on ensuring that the table replicates the structure and data of the source table. Participants then worked on each generation sequentially, answering the questionnaire after each table they worked on. In case they missed an error that was crucial to the correctness of the table while fixing the table (e.g., a missing row), the study administrator hinted at this issue to the participant. Participants completed working on generations by both TEN and TENCOT for each table they worked on before moving on to the next table.

Data Collection: We collected both quantitative (questionnaire responses) and qualitative (video and audio recordings of study sessions) data from our study. To gain a deeper understanding of the rationale behind participants' responses, we qualitatively coded the actions they performed while fixing the regenerated tables based on the study recordings. Table 5 displays the list of such actions. If many actions were performed repetitively, only one such action was coded (e.g., if participants had to add spaces between words across all row headers of the table, this action was only coded once).

Limitations: Our study only uses spreadsheets related to financial data, so our results might not generalize to other domains. Additionally, while we aimed to recruit a diverse set of participants, we use a relatively small sample size ($N = 21$), which may not be representative of all spreadsheet users. Future work could explore the usefulness of TEN in other contexts and with larger sample sizes. Further, since participants were unfamiliar with the data in the source images and did not have a personal stake in ensuring that the replication was accurate to the source data, this may have influenced their actions and responses. Finally, since both tools in the study were developed by our research team, this may lead to confirmation biases in the analysis. To mitigate this bias, we do not analyse absolute values and only report comparative results.

Code	Description
Action Type	
Add	Adding values to the replication that were present in the source table but missing from the replication
Modify	Fixing incomplete or incorrect values within individual cells in the replication
Delete	Removing values from the replication that were not present in the source table
Realign	Shifting cells that were incorrectly aligned in the replication
Data Type	
Column Heading	Heading for a column
Row Heading	Heading for a row
Subheading	Heading not for a particular row or column but for a subsection of the spreadsheet
Note	Values that are not associated with a particular row or column, but describe the data in the spreadsheet (e.g., 'values in thousands')
Data	All non-header, non-note values

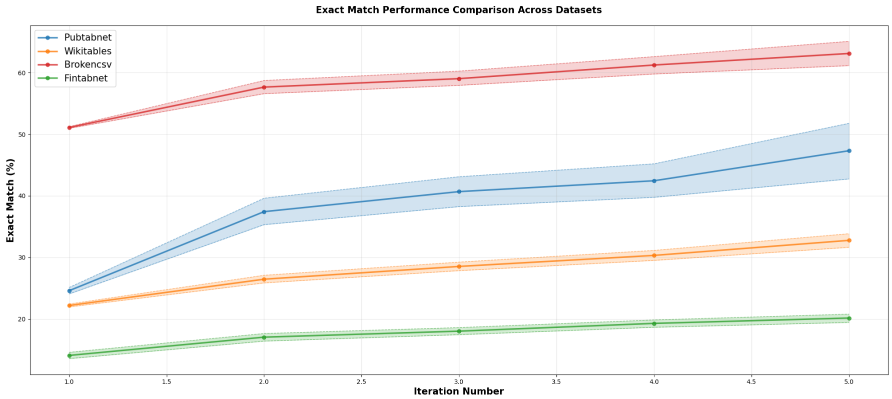
Table 5: Actions performed by user study participants to fix table regenerations. Each action consisted of an action type as well as the data type upon which the action was performed

C Results and Analysis

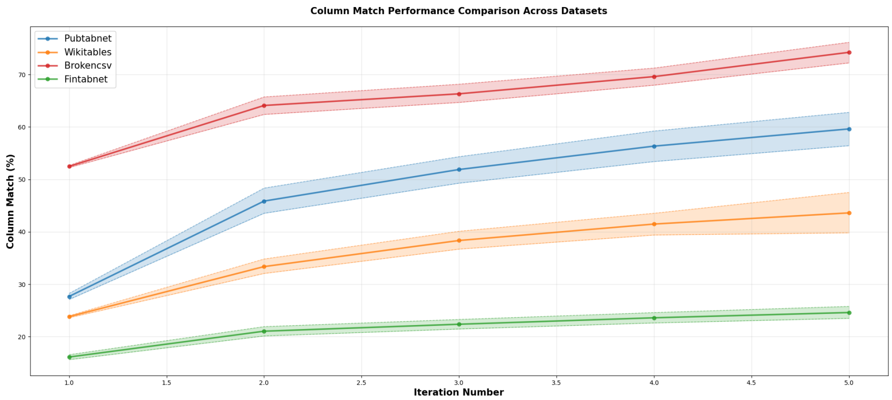
RQ1: *To what degree do prompting strategies and model choice influence table-extraction performance when evaluated on datasets of escalating complexity?*

"Structure first" beats "reason more": We evaluated the Structural Decomposition prompt (prioritizing schema/layout followed by value filling) across four datasets and six models, comparing it against a baseline prompt and Chain-of-Thought (CoT) prompting. Results (Figure 4 demonstrate that the Structural Decomposition approach consistently outperforms both baseline and CoT methods, with improvements of 10–40 EM points (e.g., PubTabNet and BrokenCSV datasets) and 8–15 points in ColVM/CVM. While CoT offers marginal gains over the baseline, the structure-first strategy yields substantially greater enhancements in table extraction accuracy.

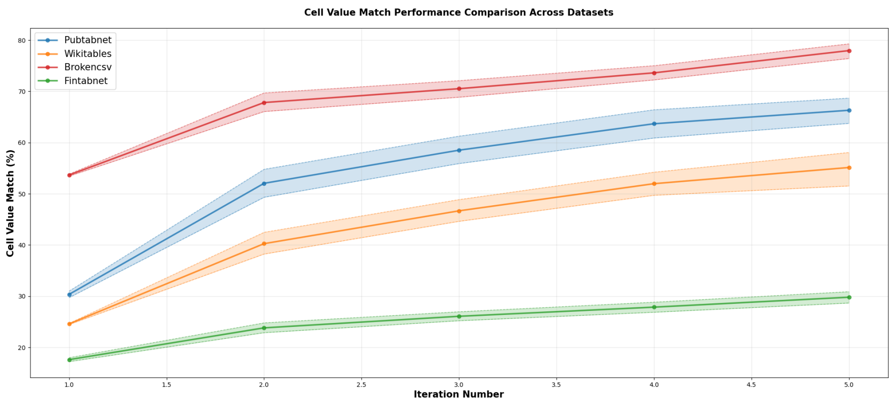
Model Capacity Matters: Model capacity plays a clear role in determining table extraction accuracy, with larger models such as GPT-4 and DeepSeek-V3-0324 consistently outperforming smaller models like o3-mini across all prompting strategies.Our results also reveal that Structural Decomposition substantially narrows this performance gap. Under the Base Prompt or COT strategies, smaller models often struggle with layout understanding, yielding low EM scores,for instance, o3-mini achieves only 2.2% EM on FinTabNet using the Base Prompt. With SD, however, o3-mini's EM rises to 14.4%, while GPT-4 improves from 8.9% to 21.7% (2.4×), indicating that SD disproportionately



(a) Exact Match



(b) Column Match



(c) Cell Value Match

Figure 7: Convergence behavior of the iterative table reconstruction system in TEN across PubTabNet, WikiTables, BrokenCSV, and FinTabNet. Most gains in Exact Match, Column Match, and Cell Value Match metrics are observed in the first 2–3 iterations, with BrokenCSV and PubTabNet achieving higher final accuracies and FinTabNet saturating early.

Prompting Strategy	Model	Cov.	Hall.	TED	E.M.	C.V.M.	Col.V.M.
BASE PROMPT	GPT-4	0.87	0.13	0.53	28.30	53.98	45.89
	DeepSeek-R1	0.77	0.18	0.41	19.80	47.05	39.42
	O3-mini	0.78	0.14	0.42	3.0	50.12	45.95
	DEEPSEEK-V3-0324	0.83	0.13	0.59	31.60	58.85	50.75
	Phi-4	0.82	0.16	0.41	15.90	45.00	34.82
	Mistral-Small	0.87	0.12	0.50	13.80	49.95	39.08
CHAIN-OF-THOUGHT(COT)	GPT-4	0.87	0.13	0.52	27.30	53.41	45.09
	DEEPSEEK-V3-0324	0.89	0.12	0.59	32.00	58.71	51.11
	Phi-4	0.82	0.17	0.41	15.6	44.72	34.90
	Mistral-Small	0.86	0.13	0.46	11.3	48.37	37.97
STRUCTURAL DECOMPOSITION	GPT-4	0.94	0.11	0.65	36.80	64.75	56.90
	DeepSeek-R1	0.96	0.10	0.74	28.7	76.06	69.20
	O3-mini	0.94	0.11	0.48	23.00	50.62	40.35
	DEEPSEEK-V3-0324	0.93	0.11	0.65	35.80	63.65	55.90
	Phi-4	0.88	0.14	0.45	13.80	47.18	36.83
	Mistral-Small	0.92	0.11	0.60	29.7	60.02	49.61

Table 6: WikiTables Dataset - Baseline comparison across prompting strategies and models. Metrics: Coverage, Hallucination Rate, Tree Edit Distance(TED), Exact Match(E.M., %), Cell Value Match(C.V.M., %), Column Value Match(Col.V.M., %).

benefits lower-capacity models. This trend is consistent across datasets, with small models gaining 15–25 percentage points in EM under SD, effectively reducing their reliance on internal structural reasoning. Nevertheless, higher-capacity models still maintain an advantage, particularly in Cell Value Match (CVM) and Column Match (ColVM) accuracy, and in handling large or complex tables.

Table Size Effects and Efficiency Implications: Table extraction performance is strongly influenced by the size of the table, regardless of the prompting strategy used. As shown in Figure 5, EM scores decline noticeably as the number of rows increases, indicating that deeper tables pose greater challenges for structural reconstruction. Similarly, performance peaks at moderate column widths (approximately 6–8 columns) and deteriorates for wider tables, likely due to increased ambiguity in column alignment and value placement. While Structural Decomposition helps mitigate these effects, larger models exhibit a more gradual degradation in accuracy as table size grows, whereas smaller models tend to plateau early, particularly on wide or deep tables.

Dataset Complexity and Performance Characteristics: Our evaluation reveals interesting relationship between dataset complexity metrics and LLM performance. BrokenCSV demonstrates the highest performance across all models (61.6-76.8 range), followed by PubTabNet (40.2-61.6 range), while FinTabNet exhibits consistently poor performance (8.9-21.9 range) and WikiTables shows moderate but variable results (13.8-36.8 range). This performance hierarchy contradicts traditional complexity assessments, where FinTabNet's sophisticated financial structures and WikiTables' diverse domain coverage would typically be considered more challenging than BrokenCSV's intentionally malformed data. The observed pattern suggests that semantic complexity—characterized by contextual richness and self-descriptive content, is more predictive of LLM performance than structural complexity. BrokenCSV's superior performance can be attributed to its design philosophy of creating self-contained, descriptive entries that remain interpretable despite formatting inconsistencies. Similarly, PubTabNet's scientific tables contain extensive contextual information, including measurement units, methodological descriptions, and domain-specific terminology that facilitate content interpretation without relying on structural cues. In contrast, FinTabNet's financial tables depend heavily on precise positional relationships and abbreviated notation that lose semantic meaning when delimiters are removed, while WikiTables' encyclopedic format often assumes structural

Prompting Strategy	Model	Cov.	Hall.	TED	E.M.	C.V.M.	Col.V.M.
BASE PROMPT	GPT-4	0.94	0.14	0.07	45	64.12	54.96
	DeepSeek-R1	0.92	0.17	0.13	47.3	65.69	56.87
	O3-mini	0.92	0.16	0.17	32.7	65.54	55.12
	DEEPSEEK-V3-0324	0.96	0.17	0.07	42.4	68.93	58.08
	Phi-4	0.88	0.17	0.07	34.8	55.87	46.15
	Mistral-Small	0.90	0.14	0.06	35.4	60.39	51.59
CHAIN-OF-THOUGHT(COT)	GPT-4	0.96	0.17	0.108	42.8	63.57	54.69
	DEEPSEEK-V3-0324	0.97	0.15	0.105	44	68.63	58.17
	Phi-4	0.88	0.2	0.04	36.6	53.43	46.85
	Mistral-Small	0.88	0.17	0.04	33.8	58.25	50.96
STRUCTURAL DECOMPOSITION	GPT-4	0.97	0.15	0.22	81.6	81.63	69.45
	DeepSeek-R1	0.98	0.14	0.21	72.1	86.56	73.28
	O3-mini	0.97	0.14	0.33	68.6	64.87	54.58
	DEEPSEEK-V3-0324	0.97	0.14	0.17	76.8	76.56	64.34
	Phi-4	0.9	0.2	0.07	63.9	60.59	51.49
	Mistral-Small	0.91	0.16	0.22	60.2	70.84	59.05

Table 7: BrokenCSV Dataset - Baseline comparison across prompting strategies and models. Metrics: Coverage, Hallucination Rate, Tree Edit Distance(TED), Exact Match(E.M., %), Cell Value Match(C.V.M., %), Column Value Match(Col.V.M., %).

Prompting Strategy	Model	Cov.	Hall.	TED	E.M.	C.V.M.	Col.V.M.
BASE PROMPT	GPT-4	0.91	0.14	0.45	30	48.18	40.22
	DeepSeek-R1	0.89	0.17	0.51	30.60	55.54	44.44
	O3-mini	0.89	0.16	0.51	22.00	55.19	46.57
	DEEPSEEK-V3-0324	0.92	0.16	0.42	27.40	49.68	37.88
	Phi-4	0.85	0.2	0.36	25.80	43.95	34.38
	Mistral-Small	0.88	0.17	0.428	27.40	46.15	36.88
CHAIN-OF-THOUGHT(COT)	GPT-4	0.91	0.15	0.48	29.80	47.38	38.38
	DEEPSEEK-V3-0324	0.92	0.149	0.50	32.00	54.46	43.16
	Phi-4	0.84	0.19	0.34	22.60	40.46	32.49
	Mistral-Small	0.84	0.16	0.32	20.00	35.73	28.53
STRUCTURAL DECOMPOSITION	GPT-4	0.93	0.14	0.67	61.60	71.76	75.41
	DeepSeek-R1	0.96	0.13	0.65	50.20	64.51	57.98
	O3-mini	0.96	0.12	0.79	41.00	53.97	49.53
	DEEPSEEK-V3-0324	0.93	0.14	0.64	52.00	65.57	59.93
	Phi-4	0.88	0.18	0.43	42.00	50.26	46.45
	Mistral-Small	0.92	0.14	0.68	40.20	57.24	49.52

Table 8: PubtabNet Dataset - Baseline comparison across prompting strategies and models. Metrics: Coverage, Hallucination Rate, Tree Edit Distance(TED), Exact Match(E.M., %), Cell Value Match(C.V.M., %), Column Value Match(Col.V.M., %).

Prompting Strategy	Model	Cov.	Hall.	TED	E.M.	C.V.M.	Col.V.M
Base Prompt	GPT-4	0.89	0.19	0.13	8.9	22.47	14.45
	DeepSeek-R1	0.84	0.19	0.28	12.4	33.17	25.31
	O3-mini	0.90	0.14	0.36	2.2	40.76	33.09
	DeepSeek-V3-0324	0.91	0.18	0.16	7.7	27.34	13.81
	Phi-4	0.84	0.21	0.013	5.7	19.04	10.01
	Mistral-Small	0.87	0.18	0.13	8.2	22.91	14.66
Chain-of-Thought(CoT)	GPT-4	0.89	0.17	0.226	10.3	24.02	16.72
	DEEPSEEK-V3-0324	0.91	0.16	0.21	11.6	29.16	19.27
	Phi-4	0.80	0.21	0.077	6.5	18.80	10.27
	Mistral-Small	0.81	0.17	0.081	4.5	16.75	10.82
Structural Decomposition	GPT-4	0.91	0.15	0.47	21.7	33.36	29.77
	DeepSeek-R1	0.94	0.14	0.44	10.1	38.84	32.17
	O3-mini	0.93	0.15	0.70	14.4	27.17	25.10
	DEEPSEEK-V3-0324	0.93	0.14	0.36	21.9±0.026	31.69	28.93
	Phi-4	0.82	0.21	0.15	8.9	18.92	10.75
	Mistral-Small	0.90	0.16	0.45	20	34.01	28.69

Table 9: FinTabNet Dataset - Baseline comparison across prompting strategies and models. Metrics: Coverage, Hallucination Rate, Tree Edit Distance(TED), Exact Match(E.M., %), Cell Value Match(C.V.M., %), Column Value Match(Col.V.M., %).

organization for disambiguation.

Iterative Feedback Loop Performance Analysis: To evaluate the effectiveness of our iterative feedback loop, we analyzed performance trajectories across multiple iterations of Structural Decomposition. As shown in Figure 7, the largest gains in EM, C.V.M., and Col.V.M. occur within the first two to three iterations, with diminishing returns thereafter. Specifically, EM typically increases by 10–20 percentage points between iterations 1 and 2, and by iteration 3 or 4, most tables reach a performance plateau. By iteration 5, improvements are minimal.

Robustness & Statistical Significance: To assess the reliability of our observed improvements, we conducted statistical analyses across all datasets and prompting strategies. We used paired bootstrap resampling (B = 10,000 samples) to compute 95% confidence intervals for the difference (δ) in performance metrics, between Structural Decomposition and each baseline strategy. As shown in Figure 8, the confidence intervals for EM, CVM, and ColVM are consistently positive and do not overlap zero, indicating statistically significant gains for Structural Decomposition across all datasets. In addition, we computed Cohen's d effect sizes to quantify the magnitude of these improvements and tested their statistical significance (Figures 9 and 10). Most effect sizes for EM, CVM, and ColVM are large and significant ($p < 0.001$), while hallucination rates show minor and dataset-dependent differences.

RQ2: How accurately does Ten reconstruct tables compared to existing baselines?

To address this research question, we compare Ten with Revelio and SplitText across four benchmarks and results are shown in a Table in the main paper. Ten consistently outperforms both Revilio and SplitText on content-fidelity metrics (EM, CVM, ColVM). On PubTabNet, Ten improves EM by +9.26 pts over Revilio, together with higher CVM (+5 pt) and ColVM (+15 pt). On the more challenging FinTabNet, absolute EM is lower for all methods, but Ten still yields a +6.05 pt EM gain and higher CVM/ColVM. On WikiTables, Ten delivers smaller yet consistent improvements (+3.37 EM; +5 CVM; +5 ColVM). The largest gains appear on BrokenCSV, where Ten attains 81.6 EM (vs. 60.96 for Revilio and 53.8 for SplitText), alongside higher CVM/ColVM.

Structural alignment, measured by TED, shows mixed results. Ten achieves lower TED on FinTabNet and is substantially better than SplitText on BrokenCSV, but slightly worse than Revilio on PubTabNet and WikiTables. This pattern suggests that Ten 's structure-first decoding can sometimes nor-

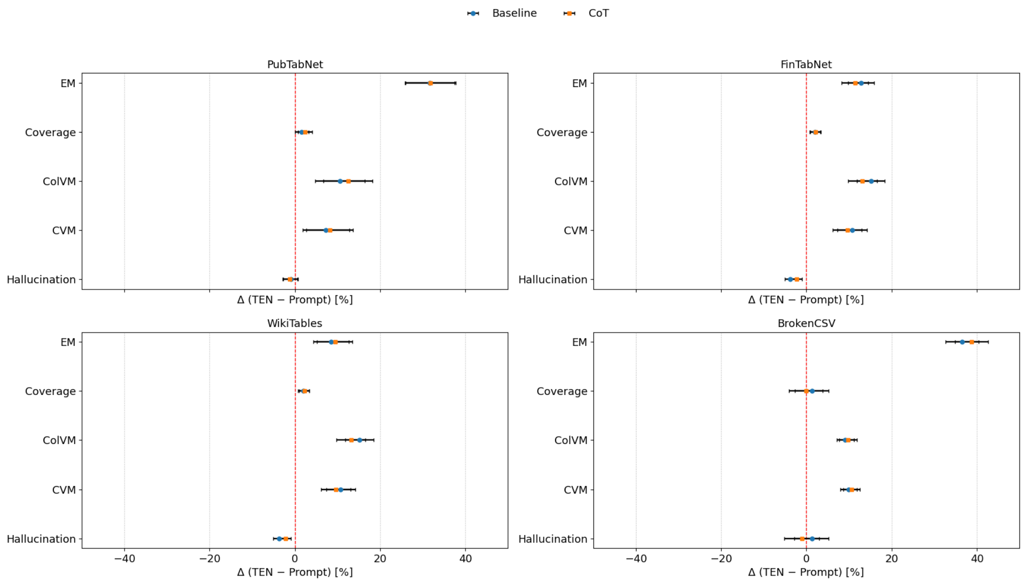


Figure 8: Absolute delta (Δ) in performance metrics between Structural Decomposition and each prompting strategy (Baseline, CoT), reported per dataset. Horizontal error bars indicate 95% bootstrap confidence intervals over 10,000 paired samples. Positive values indicate higher performance by TEN. TEN consistently improves Exact Match (EM), Column Value Match (ColVM), and Cell Value Match (CVM) across all datasets. Differences in Hallucination and Coverage vary across datasets. Significance is assessed separately in Figure.

malize header layouts or span handling in ways that improve content agreement (EM/CVM/ColVM) yet diverge modestly from the reference tree, reflecting granularity differences rather than content errors.

TED varies across datasets because it rewards structural isomorphism and penalizes benign layout normalization. It compares systems to the reference representation rather than semantic correctness. When TEN consolidates multi-row headers, propagates or flattens spans, removes empty scaffold rows/columns, or relocates footnotes and units, the values are preserved but the parse tree changes. Consequently, TED can worsen even as EM, CVM, and ColVM improve, so it should be interpreted alongside content-fidelity metrics.

RQ3: How useful is TEN at reconstructing tables from real-world financial documents?
To answer this research question, we analysed participant's actions and questionnaire responses in our user study. We compared these actions across tables generated by TEN (Structural Decomposition), hereafter referred to as TEN, and those generated by TEN (COT), hereafter referred to as TENCOT.

Perceived Accuracy: Figure 11 summarizes participants' responses to the different questions in the questionnaire provided to them. To statistically test the differences in their responses across the two conditions (TEN and TENCOT), we apply the Wilcoxon signed-rank test, as our data is paired (each participant provided one response for each condition) and ordinal (Likert scale). Further, since this test is non-parametric, it is suited to smaller sample size such as in our study ($N = 20*3 = 60$). We found that on the Likert scale from 1-7, participants reported that the accuracy of tables generated by TEN (mean = 4.97) was significantly higher than of those generated by TENCOT (mean = 4.30) (Wilcoxon signed-rank test, $W = 284.0, p = .02$) with a moderate effect size ($r = -0.30$). Further, in 7 out the 20 tables used for the study, all three reviewers independently rated TEN to be more accurate than TENCOT, while all participants preferred TENCOT for only 1 table.

Despite this difference in responses on the tables' accuracy, we did not observe any significant differences in participants' responses on the effort required to verify ($W = 260.0, p = .51$) or fix ($W = 266.0, p = .08$) these tables. Out of the 43 responses in which participants reported a difference in accuracy between tables generated by the two tools, participants did not report a corresponding difference in the effort required to fix these tables in 11 cases. Further, in the 17 responses in which participants did not report any difference

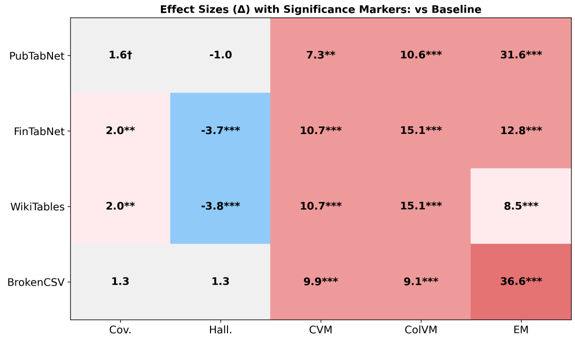


Figure 9: Heatmap showing effect sizes (Δ) and statistical significance for performance differences between Structural Decomposition and the Baseline strategy. Each cell reports the average delta ($\Delta = \text{Structural Decomposition} - \text{Baseline}$) for a given dataset and metric. Color encodes both the magnitude and direction of the effect: blue shades indicate negative effects (TEN outperforms), red shades indicate positive effects (TEN underperforms), and gray indicates non-significant results. Effect size is measured using Cohen's d; significance is marked as: *** ($p < 0.001$), ** ($p < 0.01$), † ($p < 0.1$). For EM, Coverage, ColVM, and CVM, higher values are better; for Hallucination, lower is better.

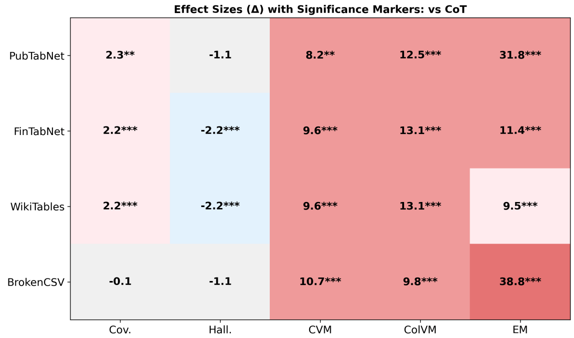


Figure 10: Heatmap showing effect sizes (Δ) and statistical significance for performance differences between Structural Decomposition and the Chain-of-Thought (CoT) prompting strategy. Each cell represents the average delta ($\Delta = \text{Structural Decomposition} - \text{Baseline}$) across datasets and metrics. Color encodes both the magnitude and direction of the effect: blue indicates negative effects (TEN outperforms CoT), red indicates positive effects (TEN underperforms), and gray marks non-significant results. Effect size is measured using Cohen's d; significance is marked as: *** ($p < 0.001$), ** ($p < 0.01$), † ($p < 0.1$). For EM, Coverage, ColVM, and CVM, higher values are better; for Hallucination, lower is better.

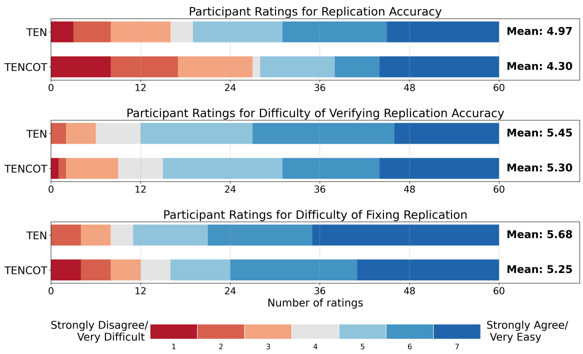


Figure 11: Questionnaire responses in user study

in accuracy between the two tools, participants reported a difference in the effort required to fix the generations by each tool in 5 cases. This suggests that accuracy alone does not fully reflect user burden and may not always be a sufficient indicator for the usefulness of the replication – the effort required to locate and fix different types of errors in these replications might play an important role in understanding the overall usability of the replication.

Participant Actions: Based on the codes in Table 5, we coded a total of 385 actions performed by participants (179 with tables generated by TEN vs 206 with tables generated by TENCOT). To assess the effect of each type of action and type of data (Table 5) on the increase in effort by participants to verify and fix the table, we perform statistical tests on those samples for which participants performed the action for one tool and did not perform the action for the other tool. Thus, for each test, one sample contains instances where participants performed the action on a particular table and one sample contains instances where participants did not perform the action for the same table (but a different tool). We do not control for which tool participants worked with in these samples to avoid any biases. While creating such pairs leads to a reduction in our sample size, this ensures that our data is paired, and allows us to continue to use the Wilcoxon signed-rank test to test for this effect. The results of these tests across different types of actions and types of data operated upon are reported in the Appendix (Section D.1). From these tests, we can infer that participants reported that it was significantly more difficult to verify the accuracy of the replication when they were required to modify values or fix misaligned data in the replication. Further, in addition to modifying values and fixing misaligned data, participants reported that it was significantly more difficult to fix replications when adding new data.

Similarly, we found that participants reported

Action Type	Avg. Difficulty With Action	Avg. Difficulty Without Action	N	W	p	r (effect size)
Add	4.71	5.66	35	64.0	0.013	-0.42
Remove	5.50	5.14	22	68.5	0.452	-0.16
Whitespace	4.67	6.22	9	0.0	0.026	-0.74
Modify	4.36	5.68	25	25.0	0.008	-0.53
Manual Misaligned	4.96	5.57	23	33.0	0.115	-0.33

Table 10: Wilcoxon signed-rank test results for fix difficulty ratings with/without performing specific action types. N represents the number of samples used for the particular test. r below

that it was significantly more difficult to fix tables when performing operations on subheadings, row headings, and data values, and reported that it was significantly more difficult to verify tables when performing operations on row headings and data values. This may be due to the fact that these types of data tend to be embedded within the table and thus may be difficult to locate and to operate on.

To analyze the occurence of the different types of actions and data that participants worked on across the two conditions, we performed McNemar's exact test (McNemar, 1947). The results of these tests are reported in the Appendix (Section D.2). We found that the number of times participants added new data to the replication was significantly higher when working on tables generated by TENCOT as compared to those generated by TEN. Further, participants operated on subheadings a significantly higher number of times when working on tables generated by TENCOT. On the other hand, the number of times participants worked on fixing misaligned data was significantly higher when working on tables generated by TEN. Upon performing the Wilcoxon signed-rank test, we found that participants rated the accuracy of replications to be significantly higher when the fixes to these replications did not require adding any data ($N = 35, W = 52.5, p = .002$) with a moderate effect size ($r = -0.53$). However, we did not find significant differences in the accuracy ratings based on fixing misaligned data in the replication ($N = 16, W = 18.0, p = .051$). [2] This suggests that participants may have reported that the tables generated by TENCOT were less accurate due to missing data in these tables. However, since both adding missing data and fixing misaligned data significantly increased the effort required to fix the replication, each tool had its own drawbacks that increased the effort for users to verify and fix the replications generated by that tool.

[2] N is lesser than 60 for these tests as we only consider those tables for which one replication required performing the action, and the other one did not.

Our results thus indicate that these different approaches to reconstructing tables might be useful in different scenarios. When accuracy is critical, applying a stricter approach that prioritizes maximum coverage (possibly at the cost of alignment issues) might ensure that the replication is useful for users. On the other hand, to ensure that the replication is easy for users to verify, we should apply a more flexible approach that ensures that individual values are accurate and correctly aligned.

RQ4: What is the impact of different components of TEN?

To assess the contributions of TEN 's feedback mechanisms, we ablated its components and evaluated four variants under identical prompts, decoding, and evaluation setup: (i) No feedback disables all feedback; (ii) Symbolic feedback enables rule-based checks; (iii) LLM feedback uses critique-and-rewrite without symbolic validation; (iv) TEN (GPT-4o) combines both. The table with these results is in the main text of the paper.

Impact of Removing Feedback Components. Compared to No feedback, Symbolic feedback consistently improves structural and content metrics, with EM gains of +5.18 (Wikipedia) and +8.62 (BrokenCSV). LLM feedback provides even larger gains on simpler or noisier datasets, e.g., +12.75 EM (Wikipedia). However, on complex tables, it is less stable and may underperform Symbolic feedback on alignment metrics.

Full System Performance. The combined TEN (GPT-4o) achieves the best performance across all metrics. It offers the highest content fidelity (EM, CVM), lowest TED, and near-saturated coverage (0.90–0.97). In some cases, LLM feedback slightly reduces hallucination compared to the full system, but this comes at the cost of structural fidelity.

Action Type	Average Fix Difficulty Rating (1-7)		N	W	p	r
	With Action	Without Action				
Add	**4.71**	**5.66**	**35**	**64.0**	**0.013**	**-0.42**
Remove	5.50	5.14	22	68.5	0.452	-0.16
Modify	**4.36**	**5.68**	**25**	**25.0**	**0.008**	**-0.53**
Realign	**4.63**	**5.81**	**16**	**13.5**	**0.022**	**-0.57**

Table 11: Wilcoxon signed-rank test results for participant ratings on effort required to fix replications with-/without performing specific action types. Participant ratings were recorded on a scale from 1 (Very Difficult) to 7 (Very Easy).

Data Type	Average Fix Difficulty Rating (1-7)		N	W	p	r
	With Data Type	Without Data Type				
Column Header	5.57	5.14	21	31.0	0.293	-0.23
Row Header	**3.76**	**5.76**	**21**	**12.5**	**0.002**	**-0.67**
Subheading	**4.68**	**5.77**	**22**	**34.5**	**0.045**	**-0.43**
Note	5.89	5.56	9	9.0	0.380	-0.29
Data	**4.17**	**6.00**	**18**	**0.0**	**0.001**	**-0.81**

Table 12: Wilcoxon signed-rank test results for participant ratings on effort required to fix replications with-/without operating on specific data types. Participant ratings were recorded on a scale from 1 (Very Difficult) to 7 (Very Easy).

Action Type	Average Verification Difficulty Rating (1-7)		N	W	p	r
	With Action	Without Action				
Add	5.11	5.51	35	70.5	0.190	-0.22
Remove	5.45	5.09	22	44.0	0.354	-0.20
Modify	**4.52**	**5.52**	**25**	**28.5**	**0.012**	**-0.50**
Realign	**4.69**	**5.56**	**16**	**5.0**	**0.036**	**-0.52**

Table 13: Wilcoxon signed-rank test results for participant ratings on verification effort with/without performing specific action types. Participant ratings were recorded on a scale from 1 (Very Difficult) to 7 (Very Easy).

Data Type	Average Verification Difficulty Rating (1-7)		N	W	p	r
	With Data Type	Without Data Type				
Column Header	5.38	5.10	21	33.0	0.373	-0.19
Row Header	**4.29**	**5.57**	**21**	**27.0**	**0.010**	**-0.56**
Subheading	5.36	5.73	22	32.5	0.357	-0.20
Note	5.44	5.44	9	10.5	1.000	0.00
Data	**4.28**	**5.44**	**18**	**10.0**	**0.012**	**-0.59**

Table 14: Wilcoxon signed-rank test results for participant ratings on verification effort with/without operating on specific data types. Participant ratings were recorded on a scale from 1 (Very Difficult) to 7 (Very Easy).

D Statistical Tests for User Study

D.1 Effect of actions on fix and verification difficulty

Tables 11 and 12 report the results of the Wilcoxon signed-rank test applied to determine the effect of performing each type of actions and of operating on each type of data respectively on participants' responses regarding the difficulty of fixing the replication. Similarly, Tables 13 and 14 report the same results for participants' responses regarding the difficulty of verifying the replication. Rows in which statistical significance was found are highlighted in bold.

Each test uses only those tables for which one replication required performing the action (or operating on the data type), and the other one did not. Thus, each test has a different sample size (N). For each test, we report the test statistic (W), p-value (p), and the effect size (r). While $|r| < 0.30$ indicates a small effect size, $0.30 \leq |r| < 0.50$ indicates a moderate effect size and $0.50 \leq |r|$ indicates a large effect size.

D.2 Occurrence of actions across TEN and TENCOT

Tables 15 and 16 report the results of applying the McNemar exact test (McNemar, 1947) across TEN and TENCOT to analyze the differences in the occurrence of each type of action and each data type in fixing the replications generated by these tools. We use the exact test as we have smaller sample sizes – the total number of pairs in our sample is $N = 60$. We report the p-values (p) and the odds ratio (OR) for each test, along with the number of occurrences for each action/data type for each tool. We calculate odds ratio as the maximum of b/c and c/b for the 2×2 contingency table:

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

E User Study Questionnaire

Following are the questions participants in our user study answered after fixing each replication. We use the Single Effort Question (SEQ) (Sauro and Dumas, 2009) to gather perceived effort to fix and verify the replication, and record responses in 7-point Likert scales.

Action	Number of occurrences in fixes		p	Odds Ratio
	TEN	TENCOT		
Add	**17**	**42**	**<0.001**	**6.00**
Remove	19	21	0.832	1.20
Modify	26	21	0.405	1.56
Realign	**40**	**28**	**0.004**	**7.00**

Table 15: McNemar exact test results for occurrence of each action.

Data Type	Number of occurrences in fixes		p	Odds Ratio
	TEN	**Baseline**		
Column Header	39	34	0.383	1.62
Row Header	18	23	0.383	1.62
Subheading	**7**	**27**	**<0.001**	**21.00**
Note	20	21	1.000	1.25
Data	18	26	0.096	2.60

Table 16: McNemar exact test results for occurrence of each data type

- Please rate your agreement with the following statement: The copy accurately replicates the structure of the original table.
  Scale: Strongly Disagree, Disagree, Somewhat Disagree, Neutral, Somewhat Agree, Agree, Strongly Agree
- Overall, how difficult or easy was it to verify the accuracy of the copy?
  Scale: Very Easy, Easy, Somewhat Easy, Neutral, Somewhat Difficult, Difficult, Very Difficult
- Overall, how difficult or easy was it to fix the copy?
  Scale: Very Easy, Easy, Somewhat Easy, Neutral, Somewhat Difficult, Difficult, Very Difficult

F Structural Decomposition Prompting Applied to PDF-Copied Table

Figure 12 displays an example of a semantically rich table containing biological annotations. When such tables are copy-pasted from a PDF, they often lose structural cues such as headers, groupings, and cell boundaries, resulting in a flattened and hard-to-interpret text block (see Listing 1). This degraded input poses a challenge for table reconstruction systems.

GO Category	Total Genes on Array	Changed Genes	Enrichment	FDR
Biological Process				
HEART				
Translation (GO:0006412)	319	32	3.334	<0.001
LIVER				
Fatty acid beta-oxidation (GO:0006635)	22	5	6.805	0.036
Translation (GO:0006412)	260	28	3.336	<0.001
Amino acid catabolic process (GO:0009063)	36	12	10.102	<0.001
Cholesterol metabolic process (GO:0008203)	54	7	4.559	0.024
Cellular respiration (GO:0045333)	47	7	4.559	0.024
Molecular Function				
HEART				
RNA Binding (GO:0003723)	455	35	2.500	<0.001
LIVER				
RNA Binding (GO:0003723)	362	28	2.326	0.001
Transaminase activity (GO:0008483)	16	10	15.428	<0.001
Oxidoreductase activity (GO:0016903)	26	6	9.281	0.001
Monooxygenase activity (GO:0004497)	98	11	3.472	0.003
Electron carrier activity (GO:0009055)	98	11	3.472	0.003

Figure 12: Original structured GO annotation table taken from PubTabNet dataset. It contains multiple biological categories (Biological Process, Molecular Function) with tissue-specific subgroups (HEART, LIVER) and associated enrichment statistics.

TEN, addresses this challenge by explicitly reconstructing the hierarchical structure of such flattened inputs. It identifies distinct categories and subgroups, organizes rows under appropriate headers, and outputs clean, semantically organized HTML tables (see Listing 2).

Listing 1 Input provided to TEN for table explicitization.

```
GO Category Total Genes on
Array
Changed
Genes Enrichment FDR
Biological Process
HEART
Translation (GO:0006412) 319 32 3.334 <0.001
LIVER
Fatty acid beta-oxidation
(GO:0006635) 22 5 6.805 0.036
Translation (GO:0006412) 260 28 3.336 <0.001
Amino acid catabolic process
(GO:0009063) 36 12 10.102 <0.001
Cholesterol metabolic process
(GO:0008203) 54 7 4.559 0.024
Cellular respiration (GO:0045333) 47 7 4.559
↪  0.024
Molecular Function
HEART
RNA Binding (GO:0003723) 455 35 2.500 <0.001
LIVER
RNA Binding (GO:0003723) 362 28 2.326 0.001
Transaminase activity
(GO:0008483) 16 10 15.428 <0.001
Oxidoreductase activity
(GO:0016903) 26 6 9.281 0.001
Monooxygenase activity
(GO:0004497) 98 11 3.472 0.003
Electron carrier activity
(GO:0009055) 98 11 3.472 0.003
```

Listing 2 Tables decomposed by TEN from structurally ambiguous input. Each table corresponds to a semantic category (e.g., Biological Process, Molecular Function) and includes GO annotations with enrichment statistics.

```json
{
  "tables": [
    {
      "starting_token": "Biological Process",
      "html_output": "<table><thead><tr><th>GO
      ↪  Category</th><th>Total Genes on
      ↪  Array</th><th>Changed Genes</th><th>Enrichment</th>
      ↪  <th>FDR</th></tr></thead><tbody><tr><td>Biological
      ↪  Process</td></tr><tr><td>HEART Translation
      ↪  (GO:0006412)</td><td>319</td><td>32</td><td>3.334</
      ↪  td><td>&lt;0.001</td></tr><tr><td>LIVER Fatty acid
      ↪  beta-oxidation
      ↪  (GO:0006635)</td><td>22</td><td>5</td><td>6.805</td
      ↪  ><td>0.036</td></tr><tr><td>Translation
      ↪  (GO:0006412)</td><td>260</td><td>28</td><td>3.336</
      ↪  td><td>&lt;0.001</td></tr><tr><td>Amino acid
      ↪  catabolic process
      ↪  (GO:0009063)</td><td>36</td><td>12</td><td>10.102</
      ↪  td><td>&lt;0.001</td></tr><tr><td>Cholesterol
      ↪  metabolic process
      ↪  (GO:0008203)</td><td>54</td><td>7</td><td>4.559</td
      ↪  ><td>0.024</td></tr><tr><td>Cellular respiration
      ↪  (GO:0045333)</td><td>47</td><td>7</td><td>4.559</td
      ↪  ><td>0.024</td></tr></tbody></table>"
    },
    {
      "starting_token": "Molecular Function",
      "html_output": "<table><thead><tr><th>GO
      ↪  Category</th><th>Total Genes on
      ↪  Array</th><th>Changed Genes</th><th>Enrichment</th>
      ↪  <th>FDR</th></tr></thead><tbody><tr><td>Molecular
      ↪  Function</td></tr><tr><td>HEART RNA Binding
      ↪  (GO:0003723)</td><td>455</td><td>35</td><td>2.500</
      ↪  td><td>&lt;0.001</td></tr><tr><td>LIVER RNA Binding
      ↪  (GO:0003723)</td><td>362</td><td>28</td><td>2.326</
      ↪  td><td>0.001</td></tr><tr><td>Transaminase activity
      ↪  (GO:0008483)</td><td>16</td><td>10</td><td>15.428</
      ↪  td><td>&lt;0.001</td></tr><tr><td>Oxidoreductase
      ↪  activity (GO:0016903)</td><td>26</td><td>6</td><td>
      ↪  9.281</td><td>0.001</td></tr><tr><td>Monooxygenase
      ↪  activity (GO:0004497)</td><td>98</td><td>11</td><td
      ↪  >3.472</td><td>0.003</td></tr><tr><td>Electron
      ↪  carrier activity
      ↪  (GO:0009055)</td><td>98</td><td>11</td><td>3.472</t
      ↪  d><td>0.003</td></tr></tbody></table>"
    }
  ]
}
```

G Prompts for Table Explicitization

G.1 Prompt Used For Structural Decomposition.

Box G.1.1: Prompt for Extracting Table

Role: You are an expert in interpreting diverse table formats. Your task is to generate structured tables from unstructured text. The primary objective is to **generalize** the process of table extraction and representation.

Instructions:

(1) Review the input text for the presence of tabular data.

(2) Identify the most appropriate row delimiter that separates the input into distinct rows. These delimiters can be system-specific (e.g., `\r\n`, `<br>`). If no such delimiter suffices, propose a custom regular expression-based delimiter and report it under `$row_delimiter$`.

(3) Identify row headers that signify the start of sub-tables or logical sections. Row headers are typically distinct lines (e.g., titles, section markers) that separate groups of rows into meaningful partitions.

(4) If row headers exist, split the table into multiple logical partitions at these row headers. Return these row headers under `$starting_token$` as partition markers.

(5) If no row headers or no multiple tables exist, treat the input as a single table. Set `$starting_token$` to `null`.

(6) For each partitioned table or the single table, generate a corresponding HTML `<table>` representation with column headers.

(7) Infer the number of columns by analyzing all rows. For rows with fewer columns, pad them with empty cells (`<td></td>` or `<th></th>`) to maintain uniformity.

(8) If a cell should span multiple rows or columns (e.g., merged cells), preserve it as-is. Do not split such cells to force a rectangular shape. Instead, pad the surrounding rows/columns appropriately to retain a rectangular layout.

(9) Wrap all column headers inside `<thead>` and all data rows (including row headers, if any) inside `<tbody>`.

(10) Do not remove, skip, or modify any content from the input text. Do not add any additional content or annotations. Do not correct spelling, formatting, or whitespace in the input.

(11) Return the final output as structured JSON, encapsulated in a code block, using the following format:

```
{
  "tables": [
    {
      "$starting_token$": "<identified_row_header_1>",
      "$html_output$": "<html_representation_of_table_1>"
    },
    ...
  ],
  "$row_delimiter$": "<row_delimiter_used>"
}
```

Input: `{{Input Text}}`

G.2 Prompt Used For Critique Generation.

Box G.2.1: Prompt used for Critique LLM

Below is a table that is constructed from a noisy text. **Role:** You are an expert in fixing messy tabular data extracted from unstructured text.
Instructions: Your task is to critically assess whether the **table is correct or messy** using your understanding of what a clean table should look like.
If you find actual structural or formatting issues, explain what they are and how to fix them. If the table structure is clear, consistent, and logical, do not suggest unnecessary changes. Prioritize semantic and visual clarity over strict rule adherence.
Do not:

(1) Add new columns or data

(2) Delete any existing columns or data

(3) Hallucinate new entries or remove original values

(4) Add indentations or extra spaces in any cells

(5) Correct spelling, reformat dates, or fill empty cells with placeholders like "N/A"

Do not change Semantics. Only the positions of content may be changed to fix structural issues.

Your critique:

G.3 Prompt Used for Table Regeneration.

Box G.3.1: Prompt for Table Regeneration Based on Structural Critique

Role: You are a table regeneration agent. **Instructions:** You will be given a previously generated table along with feedback highlighting structural or formatting issues.
Your task is to regenerate the table in **HTML format**, correcting issues such as:

- Merged cells
- Row alignment
- Header misplacement

Do Not:

- Add new rows or columns.
- Hallucinate any values.

You may split or reassign existing values across rows/columns to improve structure.

Return the regenerated table in the following structured JSON format.:

```
{
  "tables": [
    {
      "html_output": "<html_table_with_<thead>_and_<tbody>>"
    },
    ...
  ]
}
```

Critique: <Insert critique here>
Original Table: <Insert table rows here>

G.4 Prompts used for Baseline

Table extraction prompt is shown Box in G.4.1. Critique and table regeneration prompt are same as in Box G.2.1 and Box G.3.1.

Box G.4.1: Prompt for Generalized Table Extraction

Role: Role: You are an expert in interpreting diverse table formats. Your task is to generate structured tables from unstructured text. The primary objective is to **generalize** the process of table extraction and representation.
Instructions: 1. Examine the following unstructured text to identify any tabular data.
2. Convert the identified table(s) into HTML, ensuring that:

- Column headers (table headers) are wrapped within <thead></thead>.
- All data rows are wrapped within <tbody></tbody>.

3. You must return the table in structured JSON format inside code blocks as shown below.
Input:

{{Input Text}}

Output:

```
{
  "tables": [
    {
      "html_output": "<html_table_with_<thead>_and_<tbody>>"
    },
    ...
  ]}
```

First signs and symptoms, n (%)	Psoriatic spondyloarthritis		
	≤2 years (n = 51)	>10 years (n = 187)	P value
Low back pain	13 (26)	31 (17)	0.15
Sacroiliac syndrome	6 (12)	17 (9)	0.59
Neck pain	1 (2)	14 (7)	0.20
Dactylitis	5 (10)	17 (9)	0.79
Arthritis, lower limbs	29 (57)	131 (70)	0.08
Arthritis, upper limbs	27 (53)	106 (57)	0.63
Enthesitis	5 (10)	15 (8)	0.78

Figure 13: Ground truth

H Ablation Qualitative Analysis

H.1 Neurosymbolic Feedback Effectively Removes Hallucinated Columns and Restores Table Schema

In the initial output, the LLM incorrectly interpreted the label "Psoriatic spondyloarthritis" as a data column header (Figure 14), causing the insertion of an empty and unnecessary second column. This misinterpretation also flattened what was originally intended as a hierarchical two-row group header structure (Figure 13). The critique from the intermediate feedback stage correctly identified the header misalignment and flagged the redundant empty column, explicitly suggesting that "Psoriatic spondyloarthritis" should serve as a group label rather than a column header (Figure 15).

Guided by this detailed critique (Listing 1), the NeuroSymbolic loop regenerated the table by removing the hallucinated column and correctly positioning "Psoriatic spondyloarthritis" as a group-level descriptor.

```
The table extracted from the noisy text has several structural issues
      that need to be addressed for clarity and consistency. Here's a
      breakdown of the issues and how to fix them:
1. Column Headers: The first row of the table contains column headers,
      but the header for the second column ("Psoriatic
      spondyloarthritis") is not aligned with the data. It seems to be
      a category rather than a header. The actual headers should be:
   - "First signs and symptoms, n (%)"
   - "$\leq$2 years (n = 51)"
   - "$\ge$10 years (n = 187)"
   - "P value"
2. Empty Cells in the Second Column: The second column is empty for all
      rows, which suggests that the "Psoriatic spondyloarthritis" label
      might be misplaced. This column should be removed or the header
      should be integrated into the table in a way that makes sense,
      such as a title or a category label above the table.
3. Row Structure: Each row should consistently represent a symptom and
      its corresponding data across the time categories and p-value.
      The current structure is correct in terms of data placement, but
      the empty second column creates confusion.
4. Signature Mismatches: The rule-based signals indicate type
      inconsistencies, but these are primarily due to the empty second
      column and the misalignment of headers. Once the column header
      issue is resolved, these mismatches should be addressed.
To fix the table:
- Remove the second column entirely or reposition the "Psoriatic
      spondyloarthritis" label as a title above the table.
- Ensure the headers are clearly defined as:
   - "First signs and symptoms"
   - "$\leq$2 years (n = 51)"
   - "$\ge$10 years (n = 187)"
   - "P value"
- Maintain the current data structure for each symptom row, as it is
      consistent and logical once the header issue is resolved.
By addressing these issues, the table will be clearer and consistent
      without altering the original data content.
```

First signs and symptoms, n (%)	Psoriatic spondyloarthritis	≤2 years (n = 51)	>10 years (n = 187)	P value
Low back pain		13 (26)	31 (17)	0.15
Sacroiliac syndrome		6 (12)	17 (9)	0.59
Neck pain		1 (2)	14 (7)	0.20
Dactylitis		5 (10)	17 (9)	0.79
Arthritis, lower limbs		29 (57)	131 (70)	0.08
Arthritis, upper limbs		27 (53)	106 (57)	0.63
Enthesitis		5 (10)	15 (8)	0.78

Figure 14: Initial table (hallucinated column "Psoriatic spondyloarthritis")

First signs and symptoms, n (%)	≤2 years (n = 51)	>10 years (n = 187)	P value
Low back pain	13 (26)	31 (17)	0.15
Sacroiliac syndrome	6 (12)	17 (9)	0.59
Neck pain	1 (2)	14 (7)	0.20
Dactylitis	5 (10)	17 (9)	0.79
Arthritis, lower limbs	29 (57)	131 (70)	0.08
Arthritis, upper limbs	27 (53)	106 (57)	0.63
Enthesitis	5 (10)	15 (8)	0.78

Figure 15: Regenerated table (schema fixed)

Listing 1: Critique Feedback Identifying Structural Issues in the Initial Table

H.2 Comparing LLM-Only and NeuroSymbolic Feedback

In this example, the table contains hierarchical subsection labels (e.g., "Ameren", "UE", "Genco") interleaved with fuel cost data across three years (2007–2005). The initial table output failed to properly distinguish these group headers from data rows, leading to a flat and misaligned structure that impaired readability and accuracy (Figure 16a).

```
Issues Identified:
1. Inconsistent Entity Structure in Rows: The table alternates between
      headers (e.g., "UE:", "Genco:", "CILCO (AERG):", etc.) and data
      rows without clear separation or alignment. These headers are not
      properly distinguished from the data rows, making it difficult to
      interpret the table structure.
2. Misalignment of Headers and Data: The headers (e.g., "UE:", "Genco:",
      etc.) are placed in the same column as the data, which disrupts
      the logical flow of the table. Headers should ideally be
      separated or visually distinct from the data rows.
3. Inconsistent Formatting of Values: Some values include a dollar sign
      (e.g., "$1.399"), while others do not (e.g., "0.490"). This
      inconsistency can confuse readers and should be standardized.
4. Missing Column Headers for Subsections: Subsections like "UE:",
      "Genco:", "CILCO (AERG):", etc., do not have their own column
      headers, making it unclear what these sections represent. This
      lack of clarity affects the readability of the table.
5. Ambiguity in Weighted Average Rows: Rows labeled "Weighted average -
      all fuels(c)" appear multiple times under different subsections,
      but it is unclear whether these averages are specific to the
      subsection or represent a global average. This ambiguity should
      be clarified.
6. Rule-Based Signals: The flagged rule-based signals highlight
      potential type inconsistencies in various cells. While these
```

Cost of Fuels(Dollars per million Btus)	2007	2006	2005
Ameren:			
Coal^(a)	$1.399	$1.271	$1.153
Nuclear	0.490	0.434	0.421
Natural gas^(b)	7.872	8.917	9.044
Weighted average – all fuels^(c)	$1.437	$1.256	$1.184
Missouri Regulated:			
UE:			
Coal^(a)	$1.284	$1.084	$0.994
Nuclear	0.490	0.434	0.421
Natural gas^(b)	7.580	8.625	8.825
Weighted average – all fuels^(c)	$1.271	$1.035	$0.993
Non-rate-regulated Generation:			
Genco:			
Coal^(a)	$1.717	$1.691	$1.589
Natural gas^(b)	8.440	9.391	9.395
Weighted average – all fuels^(c)	$1.939	$1.865	$1.808
CILCO (AERG):			
Coal^(a)	$1.309	$1.419	$1.317
Weighted average – all fuels^(c)	$1.450	$1.466	$1.396
EEI:			
Coal^(a)	$1.329	$1.266	$1.053
Total Non-rate-regulated Generation:			
Coal^(a)	$1.545	$1.513	$1.378
Natural gas^(b)	8.440	9.385	9.384
Weighted average – all fuels^(c)	$1.698	$1.613	$1.508

(a) Ground Truth

Cost of Fuels (Dollars per million Btus)	2007	2006	2005
Ameren:			
Coal(a)	$1.399	$1.271	$1.153
Nuclear	0.490	0.434	0.421
Natural gas(b)	7.872	8.917	9.044
Weighted average – all fuels(c)	$1.437	$1.256	$1.184
Missouri Regulated:			
UE:			
Coal(a)	$1.284	$1.084	$0.994
Nuclear	0.490	0.434	0.421
Natural gas(b)	7.580	8.625	8.825
Weighted average – all fuels(c)	$1.271	$1.035	$0.993
Non-rate-regulated Generation:			
Genco:			
Coal(a)	$1.717	$1.691	$1.589
Natural gas(b)	8.440	9.391	9.395
Weighted average – all fuels(c)	$1.939	$1.865	$1.808
CILCO (AERG):			
Coal(a)	$1.309	$1.419	$1.317
Weighted average – all fuels(c)	$1.450	$1.466	$1.396
EEI:			
Coal(a)	$1.329	$1.266	$1.053
Total Non-rate-regulated Generation:			
Coal(a)	$1.545	$1.513	$1.378
Natural gas(b)	8.440	9.385	9.384
Weighted average – all fuels(c)	$1.698	$1.613	$1.508

(b) NeuroSymbolic (hierarchy preserved)

Figure 16: Comparison of outputs with NeuroSymbolic feedback vs LLM-only feedback.

```
      signals are not always accurate, they point to areas where the
      table's structure may be inconsistent or unclear.
Proposed Fixes:
To address these issues, the table should be reorganized to ensure
      structural clarity while preserving the original content and
      formatting. Below are the steps to fix the table:
1. Separate Headers from Data Rows: Move subsection headers (e.g.,
      "UE:", "Genco:", etc.) into a dedicated column to distinguish
      them from the data rows.
2. Standardize Formatting of Values: Ensure all monetary values
      consistently include a dollar sign (e.g., "$0.490" instead of
      "0.490").
3. Clarify Subsection Context: Add a column for subsection names (e.g.,
      "UE", "Genco", etc.) to make it clear which subsection each row
      belongs to.
4. Align Data Rows Under Correct Headers: Ensure that all data rows are
      properly aligned under the correct year columns (2007, 2006,
      2005).
```

Listing 2: LLM-Only Feedback Flattened the Hierarchy and Suggested Column Reorganization

The LLM-only feedback attempted to regularize this structure by converting subsection labels into a Subsection column and fuel types into a Fuel Type column (Figure 17). While this improved visual uniformity, it fundamentally altered the table's schema—flattening the hierarchy, duplicating fuel type rows, and disconnecting values from their group context. As a result, this version exhibited lower structural fidelity, despite being more readable.

```
Critique: The table extracted from the noisy text has several structural
      issues that need to be addressed for clarity and consistency.
      Here are the identified issues and suggestions for fixing them:
1. Entity Structure Consistency: The rule-based signals correctly
      identify rows where the entity structure is inconsistent. The
      table mixes headers and data rows without clear separation, which
      can confuse the reader. For example, "Ameren:" and "Missouri
      Regulated:" are headers that should be clearly distinguished from
      the data rows.
2. Header and Subheader Alignment: The headers such as "Ameren:",
      "Missouri Regulated:", "UE:", "Non-rate-regulated Generation:",
      "Genco:", "CILCO (AERG):", and "EEI:" should be aligned properly
      to indicate they are categories or subcategories. This can be
      achieved by placing them in a separate column or using
      indentation to visually separate them from the data rows.
3. Column Consistency: The table should maintain consistent column
      headers across all rows. The main header "Cost of Fuels(Dollars
      per million Btus) | 2007 | 2006 | 2005" should be clearly
      separated from the data rows, and each data row should align
      under these headers.
4. Visual Clarity: The table lacks visual clarity due to the mixing of
      headers and data. To improve this, headers should be bolded or
      separated by lines to distinguish them from the data rows.
```

Subsection	Fuel Type	2007	2006	2005
Cost of Fuels	Coal(a)	$1.399	$1.271	$1.153
Cost of Fuels	Nuclear	$0.490	$0.434	$0.421
Cost of Fuels	Natural gas(b)	$7.872	$8.917	$9.044
Cost of Fuels	Weighted average – all fuels(c)	$1.437	$1.256	$1.184
UE	Coal(a)	$1.284	$1.084	$0.994
UE	Nuclear	$0.490	$0.434	$0.421
UE	Natural gas(b)	$7.580	$8.625	$8.825
UE	Weighted average – all fuels(c)	$1.271	$1.035	$0.993
Genco	Coal(a)	$1.717	$1.691	$1.589
Genco	Natural gas(b)	$8.440	$9.391	$9.395
Genco	Weighted average – all fuels(c)	$1.939	$1.865	$1.808
CILCO (AERG)	Coal(a)	$1.309	$1.419	$1.317
CILCO (AERG)	Weighted average – all fuels(c)	$1.450	$1.466	$1.396
EEI	Coal(a)	$1.329	$1.266	$1.053
Total Non-rate-regulated	Coal(a)	$1.545	$1.513	$1.378
Total Non-rate-regulated	Natural gas(b)	$8.440	$9.385	$9.384
Total Non-rate-regulated	Weighted average – all fuels(c)	$1.698	$1.613	$1.508

Figure 17: LLM-only (flattened structure)

```
5. Grouping of Related Data: The data related to each entity (e.g.,
       Ameren, UE, Genco) should be grouped together to maintain logical
       flow. This can be done by ensuring that all rows related to a
       specific entity are contiguous and clearly marked.
6. Signature Mismatch: The signals indicate type inconsistencies, likely
       due to the mixing of headers and data. Ensuring that headers are
       not treated as data rows will resolve these mismatches.
To fix these issues, the table should be restructured to clearly
       separate headers from data rows, maintain consistent column
       alignment, and visually group related data. This will enhance
       semantic and visual clarity without altering the original text
       content.
```

Listing 3: NeuroSymbolic Feedback Identified Header Misalignment and Preserved Hierarchical Grouping

In contrast, the NeuroSymbolic feedback correctly preserved group headers as distinct non-data rows, maintained value alignment under year columns, and reduced hallucination (Figure 16). The regenerated table was semantically faithful to the original intent and achieved better alignment with the ground truth, even though minor differences in formatting remained.

Particulars	Note No.	As at March 31,2024	As at March 31,2023
A ASSETS			
Non-Current Assets			
(a) Property, Plant and Equipment	3	513.49	211.06
(b) Capital Work-in-Progress	3.1	2,847.30	871.88
(c) Other Intangible Assets	3.2	9.15	-
(d) Financial Assets			
(i) Investments in Subsidiary	4	2.00	1.80
(ii) Loans	5	183.46	53.72
(iii) Trade Receivables	6	-	11.60
(e) Other Non Current Assets	7	1,474.54	94.01
Current Assets			
(a) Inventories	8	483.99	600.89
(b) Financial Assets			
(i) Trade Receivables	9	1,342.25	647.63
(ii) Cash and Cash Equivalents	10	86.83	897.53
(iii) Loans	11	3,622.57	3,554.96
(c) Other Current Assets	12	1,040.16	625.49
Total Assets		11,605.75	7,570.58
EQUITY AND LIABILITIES			
Equity			
(a) Equity Share capital	13	1,755.47	1,668.67
(b) Other Equity	14	6,288.22	3,295.44
Total equity attributable to equity holders of the Company		8,043.69	4,964.11
LIABILITIES			
Non-Current Liabilities			
(a) Financial Liabilities			
(i) Borrowings	15	2,307.82	2,214.99
(ii) Other Financial Liabilities	16	0.25	0.85
(b) Deferred Tax Liabilities (Net)	17	12.90	3.66
Current Liabilities			
(a) Financial Liabilities			
(i) Borrowings	18	201.30	14.43
(ii) Trade Payables	19	556.20	294.25
(iii) Other Financial Liabilities	20	-	33.18
(b) Other Current Liabilities	21	388.67	19.13
(c) Provisions	22	14.63	7.67
(d) Current Tax Liabilities (Net)	23	80.28	18.30
Total Liabilities		3,562.05	2,606.47
Total Equity and Liabilities		11,605.75	7,570.58
Summary of Significant Accounting Policies	1 & 2		

Figure 18: Example of a real-world financial table from an annual report (Mercury EV – Tech Ltd., 2024) from FinRecon20 dataset. This balance sheet exhibits several structural challenges relevant to table explicitization, including hierarchical row groupings, multi-line labels, and alignment-dependent value placement.

I Qualitative Analysis of TEN's Output

This example demonstrates the table extraction and explicitization capability of TEN using a real-world financial balance sheet from Mercury EV-Tech Ltd. (Annual Report, 2024), taken from the FinReCon20 dataset. Figure 18 illustrates the original balance sheet from the financial document, exhibiting several structural complexities such as hierarchical row groupings (Assets and Liabilities), multi-line textual labels (e.g., "Property, Plant and Equipment"), and numerical values whose meaning is dependent on alignment and indentation.

I.1 Example 1

```
Particulars Note As at As at\r\nNo. March 31,2024 March 31,2023\r\nA ASSETS\r\nNon-Current Assets\r\n(a) Property, Plant and Equipment 3 513.49
    211.06\r\n(b) Capital Work-in-Progress 3.1 2,847.30 871.88\r\n(c) Other Intangible Assets 3.2 9.15 -\r\n(d) Financial Assets\r\n(i)
    Investments in Subsidiary 4 2.00 1.80\r\n(ii) Loans 5 183.46 53.72\r\n(iii) Trade Receivables 6 - 11.60\r\n(e) Other Non Current Assets 7
    1,474.54 94.01\r\n\r\nCurrent Assets\r\n(a) Inventories 8 483.99 600.89\r\n(b) Financial Assets\r\n(i) Trade Receivables 9 1,342.25 647.63\r\n(ii)
    Cash and Cash Equivalents 10 86.83 897.53\r\n(iii) Loans 11 3,622.57 3,554.96\r\n(c) Other Current Assets 12 1,040.16 625.49\r\nTotal Assets
    11,605.75 7,570.58\r\nEQUITY AND LIABILITIES\r\nEquity\r\n(a) Equity Share capital 13 1,755.47 1,668.67\r\n(b) Other Equity 14 6,288.22
    3,295.44\r\nTotal equity attributable to equity holders of the Company 8,043.69 4,964.11\r\nLIABILITIES\r\nNon-Current Liabilities\r\n(a)
    Financial Liabilities\r\n(i) Borrowings 15 2,307.82 2,214.99\r\n(ii) Other Financial Liabilities 16 0.25 0.85\r\n(b) Deferred Tax Liabilities
    (Net) 17 12.90 3.66\r\nCurrent Liabilities\r\n(a) Financial Liabilities\r\n(i) Borrowings 18 201.30 14.43\r\n(ii) Trade Payables 19 556.20
    294.25\r\n(iii) Other Financial Liabilities 20 - 33.18\r\n(b) Other Current Liabilities 21 388.67 19.13\r\n(c) Provisions 22 14.63 7.67\r\n(d)
    Current Tax Liabilities (Net) 23 80.28 18.30\r\nTotal Liabilities 3,562.05 2,606.47\r\nTotal Equity and Liabilities 11,605.75
    7,570.58\r\nSummary of Significant Accounting Policies 1 & 2
```

Listing 4: Input text to TEN: a structurally ambiguous table representation resulting from manual copy-pasting from a financial report. This serves as the source for table explicitization.

Listing 4 provides the raw, structurally ambiguous input text that results from a typical manual copy-paste operation from the PDF document. Notice that the hierarchical relationships and structural semantics of the table are lost due to flattening, inconsistent spacing, and disrupted formatting. Such ambiguities pose significant challenges for traditional extraction methods relying solely on visual layout cues. Figure 19 presents the output produced by TEN. The system successfully reconstructs the table structure, accurately identifying and restoring:

- Multi-level hierarchical relationships (e.g., clearly distinguishing between Non-Current and Current Assets and Liabilities).
- Complex, multi-line row labels, correctly segmented and aligned.
- Numerical value placements matching the original structural semantics of the balance sheet.

This demonstrates the effectiveness of TEN in recovering structured representations from highly ambiguous, flattened textual inputs, significantly reducing manual table-reconstruction effort for end-users.

I.2 Example 2

This example showcases the reconstruction performance of TEN on a challenging, multi-segment revenue breakdown table from a corporate annual report. The ground-truth table (Figure 20) captures revenues segmented geographically (Americas, Europe, Asia Pacific) and by business solutions (Merchant, Issuer, Consumer Solutions) across three consecutive financial years (2020–2022).

Listing 5 illustrates the unstructured, copy-pasted input provided to TEN. The flattened textual representation lacks clear delineation of headers, rows, and hierarchical relationships, creating ambiguity for structural recovery.

```
YearEndedDecember31,2022\r\n MerchantSolutions\r\n Issuer\r\n Solutions ConsumerSolutions\r\n Intersegment\r\n Eliminations Total\r\n
      (inthousands)\r\n Americas $5,236,728 $1,739,620 $620,482 $(58,916) $7,537,914\r\n Europe 720,660 469,412 - - 1,190,072\r\n AsiaPacific 247,529
      36,591 - (36,591) 247,529\r\n $6,204,917 $2,245,623 $620,482 $(95,507) $8,975,515\r\n YearEndedDecember31,2021\r\n MerchantSolutions\r\n
      Issuer\r\n Solutions ConsumerSolutions\r\n Intersegment\r\n Eliminations Total\r\n (inthousands)\r\n Americas $4,735,505 $1,644,765 $783,625
      $(65,781) $7,098,114\r\n Europe 684,760 495,597 - - 1,180,357\r\n AsiaPacific 245,292 25,385 - (25,386) 245,291\r\n $5,665,557 $2,165,747
      $783,625 $(91,167) $8,523,762\r\n YearEndedDecember31,2020\r\n MerchantSolutions\r\n Issuer\r\n Solutions ConsumerSolutions\r\n Intersegment\r\n
      Eliminations Total\r\n (inthousands)\r\n Americas $3,948,643 $1,601,118 $747,886 $(64,308) $6,233,339\r\n Europe 539,838 450,529 - - 990,367\r\n
      AsiaPacific 199,854 9,725 - (9,727) 199,852\r\n $4,688,335 $2,061,372 $747,886 $(74,035) $7,423,558\r\n
```

Listing 5: Unstructured input from multi-segment revenue breakdown table, used as input to TEN

The resulting structured table generated by TEN is presented in Figure 21. Notably, TEN successfully:

- Preserves the core temporal structure, clearly segmenting data by year.
- Maintains geographic and business segment distinctions, correctly recovering the hierarchical table layout.
- Accurately recovers and aligns numerical values in the majority of cases.
- Appropriately isolates and separates the distribution channel breakdown, maintaining its distinction from the main revenue segment.

However, two primary reconstruction issues emerge:

1. **Hallucinated text entries (highlighted in red):** TEN introduces textual content not present in the source input, reflecting challenges in fully resolving ambiguous textual labels.
2. **Misalignment of numerical values (highlighted in orange):** Certain numerical data entries are incorrectly positioned, resulting in mismatches with their respective column headers or row categories.

I.3 Example 3

This example illustrates the performance of TEN on reconstructing a structurally intricate multi-year financial performance report extracted from a corporate financial statement. The original table (Figure 22) captures detailed financial metrics, including average balances, interest earned, and yield rates across multiple asset and liability classes spanning three consecutive years (2021–2023). Notable complexities in this table include:

Particulars	Note No.	As at March 31,2024	As at March 31,2023
A ASSETS			
Non-Current Assets			
(a) Property, Plant and Equipment	3	513.49	211.06
(b) Capital Work-in-Progress	3.1	2,847.30	871.88
(c) Other Intangible Assets	3.2	9.15	-
(d) Financial Assets			
(i) Investments in Subsidiary	4	2.00	1.80
(ii) Loans	5	183.46	53.72
(iii) Trade Receivables	6	-	11.60
(e) Other Non Current Assets	7	1,474.54	94.01
Current Assets			
(a) Inventories	8	483.99	600.89
(b) Financial Assets			
(i) Trade Receivables	9	1,342.25	647.63
(ii) Cash and Cash Equivalents	10	86.83	897.53
(iii) Loans	11	3,622.57	3,554.96
(c) Other Current Assets	12	1,040.16	625.49
Total Assets		11,605.75	7,570.58
EQUITY AND LIABILITIES			
Equity			
(a) Equity Share capital	13	1,755.47	1,668.67
(b) Other Equity	14	6,288.22	3,295.44
Total equity attributable to equity holders of the Company		8,043.69	4,964.11
LIABILITIES			
Non-Current Liabilities			
(a) Financial Liabilities			
(i) Borrowings	15	2,307.82	2,214.99
(ii) Other Financial Liabilities	16	0.25	0.85
(b) Deferred Tax Liabilities (Net)	17	12.90	3.66
Current Liabilities			
(a) Financial Liabilities			
(i) Borrowings	18	201.30	14.43
(ii) Trade Payables	19	556.20	294.25
(iii) Other Financial Liabilities	20	-	33.18
(b) Other Current Liabilities	21	388.67	19.13
(c) Provisions	22	14.63	7.67
(d) Current Tax Liabilities (Net)	23	80.28	18.30
Total Liabilities		3,562.05	2,606.47
Total Equity and Liabilities		11,605.75	7,570.58
Summary of Significant Accounting Policies	1 & 2		

Figure 19: Structured balance sheet generated by the TEN from unstructured copy-paste input. TEN accurately recovers multi-level headings, financial hierarchies, and aligned numerical values, closely matching the original document layout.

	Year Ended December 31, 2022				
	Merchant Solutions	Issuer Solutions	Consumer Solutions	Intersegment Eliminations	Total
			(in thousands)		
Americas	$5,236,728	$1,739,620	$620,482	$(58,916)	$7,537,914
Europe	720,660	469,412	—	—	1,190,072
Asia Pacific	247,529	36,591	—	(36,591)	247,529
	$6,204,917	$2,245,623	$620,482	$(95,507)	$8,975,515

	Year Ended December 31, 2021				
	Merchant Solutions	Issuer Solutions	Consumer Solutions	Intersegment Eliminations	Total
			(in thousands)		
Americas	$4,735,505	$1,644,765	$783,625	$(65,781)	$7,098,114
Europe	684,760	495,597	—	—	1,180,357
Asia Pacific	245,292	25,385	—	(25,386)	245,291
	$5,665,557	$2,165,747	$783,625	$(91,167)	$8,523,762

	Year Ended December 31, 2020				
	Merchant Solutions	Issuer Solutions	Consumer Solutions	Intersegment Eliminations	Total
			(in thousands)		
Americas	$3,948,643	$1,601,118	$747,886	$(64,308)	$6,233,339
Europe	539,838	450,529	—	—	990,367
Asia Pacific	199,854	9,725	—	(9,727)	199,852
	$4,688,335	$2,061,372	$747,886	$(74,035)	$7,423,558

The following table presents a disaggregation of our Merchant Solutions segment revenues by distribution channel for the years ended December 31, 2022, 2021 and 2020:

	2022	2021	2020
		(in thousands)	
Relationship-led	$3,189,046	$3,031,873	$2,600,440
Technology-enabled	3,015,871	2,633,684	2,087,895
	$6,204,917	$5,665,557	$4,688,335

Figure 20: Ground truth table of segment-wise revenues by geography and product line, extracted from a corporate annual report. The table spans three years (2020–2022) and contains five financial dimensions: Merchant Solutions, Issuer Solutions, Consumer Solutions, Intersegment Eliminations, and Totals. Complexities include repeated multi-level headers, hierarchical geographies, and negative values, all of which challenge structure recovery in copy-pasted representations.

Region	Merchant Solutions	Issuer Solutions	Consumer Solutions	Intersegment Eliminations	Total
(in thousands)					
Americas	$5,236,728	$1,739,620	$620,482	$(58,916)	$7,537,914
Europe	$720,660	$469,412	—	—	$1,190,072
AsiaPacific	$247,529	$36,591	—	$(36,591)	$247,529
Total	$6,204,917	$2,245,623	$620,482	$(95,507)	$8,975,515
Region	**Merchant Solutions**	**Issuer Solutions**	**Consumer Solutions**	**Intersegment Eliminations**	**Total**
(in thousands)					
Americas	$4,735,505	$1,644,765	$783,625	$(65,781)	$7,098,114
Europe	$684,760	$495,597	—	—	$1,180,357
AsiaPacific	$245,292	$25,385	—	$(25,386)	$245,291
Total	$5,665,557	$2,165,747	$783,625	$(91,167)	$8,523,762
Region	**Merchant Solutions**	**Issuer Solutions**	**Consumer Solutions**	**Intersegment Eliminations**	**Total**
(in thousands)					
Americas	$3,948,643	$1,601,118	$747,886	$(64,308)	$6,233,339
Europe	$539,838	$450,529	—	—	$990,367
AsiaPacific	$199,854	$9,725	—	$(9,727)	$199,852
Total	$4,688,335	$2,061,372	$747,886	$(74,035)	$7,423,558

Type	2022	2021	2020
(in thousands)			
Relationship-led	$3,189,046	$3,031,873	$2,600,440
Technology-enabled	$3,015,871	$2,633,684	$2,087,895
Total	$6,204,917	$5,665,557	$4,688,335

Figure 21: Table generated by TEN. TEN successfully preserved the core hierarchical structure with three distinct temporal sections and maintained geographic segmentation across most business segments. TEN is also able to preserve numerical data across the majority of cells and successful separation of the distribution channel breakdown table as a distinct section. However, there are also some text-based reconstruction issues: (1) hallucinated text entries (highlighted in red) where TEN introduced content not present in the original document, and (2) misalignment issues (highlighted in light orange) where data values from the source were incorrectly positioned, resulting in values being associated with wrong column headers or row categories.

- Multi-level column headers, with repeated labels for each year (e.g.,“Average Balance”, “Interest”, “Yield”).
- Nested row labels categorizing financial elements such as assets, liabilities, and equity.
- Numerical values in varying formats, including percentages, currency, and absolute figures.

Listing 6 shows the raw, structurally ambiguous textual input provided to TEN. This flattened textual representation features challenges such as inconsistent whitespace, merged and misaligned columns, and disrupted multi-line headers, representing typical copy-paste noise encountered in practice.

The structured table generated by TEN (Figure 23) demonstrates significant capability in:

- Accurately reconstructing the overall table layout, successfully segmenting yearly data and maintaining clear distinctions between asset and liability categories.
- Preserving hierarchical header structures and correctly associating most numerical values with their corresponding financial dimensions.

However, this example also highlights specific reconstruction errors:

1. **Hallucinated numerical values (highlighted in red):** TEN incorrectly introduces numeric data absent from the original source, indicating difficulties in interpreting ambiguous textual inputs.
2. **Misaligned numerical values (highlighted in yellow):** Although the data values themselves are correctly extracted from the source, they are occasionally misplaced into incorrect cells due to confusion between repeated sub-headers or loss of spatial alignment cues inherent in the original visual representation.

These observations underscore existing limitations of TEN when reconstructing structurally complex tables relying exclusively on noisy textual inputs without visual formatting cues.

```
Year Ended December 31, 2023 Year Ended December 31, 2022 Year Ended December 31, 2021 \r\n Average Interest Average Interest Average Interest \r\n
       Daily and Yield/ Daily and Yield/ Daily and Yield/ \r\n Balance Dividends Cost(4) Balance Dividends Cost(4) Balance Dividends Cost(4) \r\n
       (Dollars in Thousands) \r\nAssets: \r\nInterest earning assets: \r\nInvestment securities $ 328,533 $ 11,376 3.46% $ 336,779 $ 8,579 2.55% $
       215,978 $ 4,238 1.96% \r\nFHLB and FRB stock 12,851 727 5.66 6,369 302 4.74 4,831 255 5.28 \r\nLoans receivable(1) 1,436,672 79,423 5.53
       1,194,788 60,353 5.05 914,804 45,134 4.93 \r\nOther earning assets 2,671 89 3.33 34,170 228 0.67 74,102 120 0.16 \r\nTotal interest earning
       assets 1,780,727 91,615 5.14 1,572,106 69,462 4.42 1,209,715 49,747 4.11 \r\nNoninterest earning assets 234,859 196,813 147,534 \r\nTotal assets
       $2,015,586 $1,768,919 $1,357,249 \r\n \r\nLiabilities and equity: \r\nInterest-bearing liabilities: \r\nDeposit accounts: \r\nChecking $ 237,006
       $ 595 0.25% $ 244,208 $ 173 0.07% $ 190,645 $ 47 0.02% \r\nSavings 238,695 146 0.06 269,033 128 0.05 198,648 117 0.06 \r\nMoney market 331,199
       5,548 1.68 358,122 1,711 0.48 244,113 545 0.22 \r\nCertificates of deposit 357,573 11,568 3.24 188,954 1,112 0.59 158,959 765 0.48 \r\nFHLB
       advances and other borrowings 159,667 8,562 5.36 14,627 514 3.51 9,411 175 1.86 \r\nOther long-term debt 58,930 2,719 4.61 59,807 2,512 4.2
       29,834 1,558 5.22 \r\nTotal interest-bearing liabilities 1,383,070 29,138 2.11 1,134,751 6,150 0.54 831,610 3,207 0.39 \r\nNoninterest checking
       439,388 453,841 346,243 \r\nOther noninterest-bearing liabilities 34,321 24,672 22,382 \r\nTotal liabilities 1,856,779 1,613,264 1,200,235 \r\n
       \r\nTotal equity 158,807 155,655 157,014 \r\n \r\nTotal liabilities and equity $2,015,586 $1,768,919 $1,357,249 \r\nNet interest income/interest
       rate spread(2) $ 62,477 3.04% $ 63,312 3.88% $ 46,540 3.72% \r\n \r\nNet interest margin(3) 3.51% 4.03% 3.85% \r\nTotal interest earning assets
       to interest\r\nbearing liabilities 128.75% 138.54% 145.47%
```

Listing 6: Input text to TEN: a structurally ambiguous table representation resulting from manual copy-pasting from a financial report. This unstructured text contains noise such as misaligned columns, merged headings, and inconsistent whitespace, and serves as the input for table explicitization.

	Year Ended December 31, 2023			Year Ended December 31, 2022			Year Ended December 31, 2021		
	Average Daily Balance	Interest and Dividends	Yield/ Cost(4)	Average Daily Balance	Interest and Dividends	Yield/ Cost(4)	Average Daily Balance	Interest and Dividends	Yield/ Cost(4)
				(Dollars in Thousands)					
Assets:									
Interest earning assets:									
Investment securities	$ 328,533	$ 11,376	3.46%	$ 336,779	$ 8,579	2.55%	$ 215,978	$ 4,238	1.96%
FHLB and FRB stock	12,851	727	5.66	6,369	302	4.74	4,831	255	5.28
Loans receivable(1)	1,436,672	79,423	5.53	1,194,788	60,353	5.05	914,804	45,134	4.93
Other earning assets	2,671	89	3.33	34,170	228	0.67	74,102	120	0.16
Total interest earning assets	1,780,727	91,615	5.14	1,572,106	69,462	4.42	1,209,715	49,747	4.11
Noninterest earning assets	234,859			196,813			147,534		
Total assets	$2,015,586			$1,768,919			$1,357,249		
Liabilities and equity:									
Interest-bearing liabilities:									
Deposit accounts:									
Checking	$ 237,006	$ 595	0.25%	$ 244,208	$ 173	0.07%	$ 190,645	$ 47	0.02%
Savings	238,695	146	0.06	269,033	128	0.05	198,648	117	0.06
Money market	331,199	5,548	1.68	358,122	1,711	0.48	244,113	545	0.22
Certificates of deposit	357,573	11,568	3.24	188,954	1,112	0.59	158,959	765	0.48
FHLB advances and other borrowings	159,667	8,562	5.36	14,627	514	3.51	9,411	175	1.86
Other long-term debt	58,930	2,719	4.61	59,807	2,512	4.2	29,834	1,558	5.22
Total interest-bearing liabilities	1,383,070	29,138	2.11	1,134,751	6,150	0.54	831,610	3,207	0.39
Noninterest checking	439,388			453,841			346,243		
Other noninterest-bearing liabilities	34,321			24,672			22,382		
Total liabilities	1,856,779			1,613,264			1,200,235		
Total equity	158,807			155,655			157,014		
Total liabilities and equity	$2,015,586			$1,768,919			$1,357,249		
Net interest income/interest rate spread(2)		$ 62,477	3.04%		$ 63,312	3.88%		$ 46,540	3.72%
Net interest margin(3)			3.51%			4.03%			3.85%
Total interest earning assets to interest-bearing liabilities			128.75%			138.54%			145.47%

Figure 22: The table presents a multi-year financial performance report, including average balances, interest earned, and yields across assets and liabilities. The original table has multi-level column headers and repeated metrics across years.

Category	Year Ended December 31, 2023	Year Ended December 31, 2022	Year Ended December 31, 2021						
Average Daily Balance	Interest/Dividends	Yield/Cost(4)	Average Daily Balance	Interest/Dividends	Yield/Cost(4)	Average Daily Balance	Interest/Dividends	Yield/Cost(4)	
(Dollars in Thousands)									
Assets:									
Interest earning assets:									
Investment securities	328,533	11,376	3.46%	336,779	8,579	2.55%	215,978	4,238	1.96%
FHLB and FRB stock	12,851	727	5.66	6,369	302	4.74	4,831	255	5.28
Loans receivable(1)	1,436,672	79,423	5.53	1,194,788	60,353	5.05	914,804	45,134	4.93
Other earning assets	2,671	89	3.33	34,170	228	0.67	74,102	120	0.16
Total interest earning assets	1,780,727	91,615	5.14	1,572,106	69,462	4.42	1,209,715	49,747	4.11
Noninterest earning assets	234,859		196,813		147,534				
Total assets	$2,015,586		$1,768,919		$1,357,249				
Liabilities and equity:									
Interest-bearing liabilities:									
Deposit accounts:									
Checking	237,006	595	0.25%	244,208	173	0.07%	190,645	47	0.02%
Savings	238,695	146	0.06	269,033	128	0.05	198,648	117	0.06
Money market	331,199	5,548	1.68	358,122	1,711	0.48	244,113	545	0.22
Certificates of deposit	357,573	11,568	3.24	188,954	1,112	0.59	158,959	765	0.48
FHLB advances and other borrowings	159,667	8,562	5.36	14,627	514	3.51	9,411	175	1.86
Other long-term debt	58,930	2,719	4.61	59,807	2,512	4.2	29,834	1,558	5.22
Total interest-bearing liabilities	1,383,070	29,138	2.11	1,134,751	6,150	0.54	831,610	3,207	0.39
Noninterest checking	439,388		453,841		346,243				
Other noninterest-bearing liabilities	34,321		24,672		22,382				
Total liabilities	1,856,779		1,613,264		1,200,235				
Total equity	158,807		155,655		157,014				
Total liabilities and equity	$2,015,586		$1,768,919		$1,357,249				
Net interest income/interest rate spread(2)	$ 62,477	3.04%		$ 63,312	3.88%		$ 46,540	3.72%	
Net interest margin(3)	3.51%	4.03%	3.85%						
Total interest earning assets to interest-bearing liabilities	128.75%	138.54%	145.47%						

Figure 23: Table generated by TEN. Despite the structural intricacy of the table, featuring nested header rows, label repetitions across columns, and varying numerical formats, TEN is able to successfully reconstruct the overall layout and preserve the majority of header hierarchies and cell alignments with high fidelity. Cells shaded in light red represent *hallucinated values*, which were not present in the ground truth but were incorrectly introduced and yellow cells indicate *misaligned values*, where content was correctly extracted but inserted into an incorrect row or column, typically due to confusion between repeated sub-headers. These errors highlight current limitations of TEN in aligning hierarchical headers and preserving structural context, especially when operating solely on text extracted via copy-paste. In such scenarios, visual layout cues such as merged cells, indentation, and spacing are lost, making it difficult for language models to infer correct associations between headers and values.